Universitat de Lleida
Escola Politècnica Superior
Enginyeria Tècnica en Informàtica de Sistemes

Treball de final de carrera

Avaluació de la xarxa de comunicacions de la Universitat de Lleida mitjançant tecnologia Honeypot

Autor: Francesc Guitart Bravo

Director: Carles Mateu Piñol

Juny de 2007

Índex

Índex de figures

Índex de taules

Capítol 1

Pròleg

Aquest projecte neix al juny del 2006 com idea d'en Carles Mateu. L'Enric Guitart m'ho va proposar a mi, encoratjant-me i fent-me'l atractiu. Molt il·lusionat en aquell moment vaig dur a terme la fase de documentació, que em va protar del juliol fins al novembre del 2006 llegint i buscant articles, papers i llibres sobre els honeypot. Mai hagués pensat que la informàtica pogués ser tan humana.

Al novembre vaig poder començar a fer proves en l'aula Alcatel de l'Escola Politècnica Superior, on després d'hores i hores vaig poder construir dos honeypot: l'Arale i en Senbei. A l'abril del 2007 vaig començar a capturar els primers atacs.

L'estudi d'aquestes eines m'ha proporcionat molts coneixements en el camp de les comunicacions, coneixements que de ben segur no oblidaré. Aquest document només és una petita mostra de l'experiencia treballada, la resta queda per mi.

La gratitut no és quelcom que es pugui plasmar en un full de paper, tot i així vull agrair en primer lloc a l'Enric, el meu pare, ja que sense ell res d'això hagués estat possible. Agraeixo també al Carles el seu suport i la confiança dipositada en mi. Encara que no ho vulguis gràcies mare per tot, també gràcies a tu Enric. Si de vegades es díficil aguantar a un informàtic, no vull imaginar aguantar-ne a dos. Gràcies a l'àrea CCIA (Ciències de la Compu-

tació i Intel·ligència Artificial) de la EPS per deixar-me l'Arale i en Senbei, els trobaré a faltar. I per últim, gràcies als meus companys per soportar les historietes de guerra en els meus honeypots durant els interminables cafés i per confiar en mi quan res funcionava i se m'acabava la fe.

Capítol 2

Introducció

Des que el 1972, aparegueren les primeres demostracions de l'ARPANET [1], fins avui dia que es calcula que la xarxa mundial de xarxes sobrepassa els mil cent milions d'usuaris, les comunicacions han sofert grans canvis, tant des del punt de vista tecnològic com del sociocultural. Tant és així, que hem arribat a introduir aquestes tecnologies en les nostres vides fent-les part del que anomenem quotidià.

Actualment hi ha xarxes de comunicacions de tota mena en qualsevol lloc. Des de petites xarxes locals, com la que pot tenir un usuari qualsevol a casa seva, interconnectant poques màquines, fins grans xarxes corporatives com les que podem trobar en empreses i universitats. En tots dos casos, s'utilitzen vies de comunicació per al transport de la informació, la qual esdevé més o menys valuosa segons la xarxa de que es tracta.

D'aquesta manera veiem que en una gran xarxa d'una empresa determinada, la informació continguda és de rellevant importància per motius obvis, i aquest fet crida l'atenció de persones alienes a la corporació fins al punt que poden voler violar tots els sistemes de seguretat per tal d'accedir aquesta informació.

[1] La Advanced Research Projects Agency Network (ARPANET) desenvolupada per ARPA del Departament de Defensa dels Estats Units, va ser la primera *packet switching network* operacional, i la progenitora de Internet

El *Honeynet Project* [7] ens aporta algunes dades interessants, per exemple:

- Un ordinador d'internet es escanejat dotze vegades al dia.
- El temps que algú triga en entrar il·lícitament en un servidor amb la instal·lació bàsica de RedHat 6.2 és de menys de 72 hores.
- Un ordinador domèstic funcionant sobre Windows 98 i amb la compartició de fitxers habilitada va ser vulnerat 4 vegades en 5 dies.
- El rècord d'accés a un servidor acabat de connectar a internet és de 15 minuts després de la connexió.

Aquests fets evidencien la necessitat de què qualsevol entitat que valori les tecnologies de la informació, ha d'estar a la vanguàrdia en sistemes de seguretat per ser conscients dels riscos que s'assumeixen en utilitzar les xarxes de comunicacions.

En aspectes de seguretat, la batalla dels administradors deixa de quedar-se només en un pla defensiu, evitant atacs i reparant els desperfectes produïts per possibles intrusions. La postura proactiva de quedar-se a la defensiva envers els atacs deixa sempre la víctima mal parada i en una situació de desavantatge en haver de veure permanentment la xarxa en producció en perill.

Els Honeypots han demostrat capgirar la situació, ja que cada vegada més, tant empreses com institucions públiques i privades, uneixen esforços per tal d'aturar la delinqüència telemàtica parant paranys per aprendre les tècniques i tàctiques dels enemics. Així doncs, sembla que els Honeypots poden desplaçar la guerra entre administradors i delinqüents cap a un altre lloc, ben lluny de la informació que emmagatzemen les nostres màquines.

2.1 L'Enemic

A primer cop d'ull, i fent un abús del terme, podriem dir, que el nostre enemic és un *hacker* que vol robar informació valuosa per a fins obscurs. Si fem

aquesta generalització segurament ens estarem equivocant tant en qui és el nostre enemic com en els seus fins. El primer error que cometem és en titllar de *hacker* a tot aquell que intenta fer una intrusió a la nostra xarxa. La paraula *hacker* ha estat maltractada, sobretot en termes de les comunicacions, ja que *hacker* és vist com un enemic de la seguretat informàtica.

L'etimologia de la paraula, sorgida del mot *to hack* de l'anglès que vol dir "tallar a cops de destral", va començar a ser utilitzada en el món de la informàtica en els anys 60, en l'entorn del MIT (*Massachusetts Institute of Technology*), on un *hacker* era aquell qui utilitzant l'ingeni millorava quelcom (habitualment programes informàtics). En l'entorn que es va desenvolupar, està clar que havia d'acabar essent una paraula fortament lligada a la informàtica i les telecomunicacions, però no només es restringeix a aquest àmbit. El mateix Richard Stallman diu:"*Playfully doing something difficult, whether useful or not, that is hacking*". Per tant qualsevol persona del món és un hacker en potència, fins i tot sense haver vist un ordinador en la vida i el més significant és que no ha de ser malèvol.

Molts autors i administradors informàtics han intentat crear dos estereotips d'atacants diferenciant-los per la seva habilitat o el seu nivell informàtic. Així doncs ens podem trobar amb els anomenats "script kiddies", que utilitzen guions, programes, i altres eines que troben per internet, intentant entrar en equips sense saber realment a què s'estan enfrontant. D'aquí el seu nom, nen de guió. El resultat és sovint atacs estúpids per força bruta, intents d'atac inútils contra serveis inexistents i atacs automatitzats, això fa que sigui l'atac més comú dins de la xarxa de xarxes. Els seus objectius són generalment ordinadors al atzar i pàgines web per poder demostrar el seu domini a altres *crackers* del seu nivell. Per altra banda, ens trobem atacants amb un nivell molt superior que utilitzen el seu ingeni i el seu coneixement en el medi per tal d'atacar els seus objectius. Són pràcticament indetectables, tot depenent del seu nivell i les seves motivacions. Generalment busquen informacions d'algun tipus d'organisme, banc o organització. Aquest tipus d'atacant s'apropa més a la definició de *hacker* per el seu nivell

com a informàtic, malgrat tot no deixa de ser un delinqüent.

Ara podem parlar de *crackers*, delinqüents informàtics i usuaris maliciosos, que són aquells que sota l'etiqueta de *black hat* poden convertir-se en enemics de la nostra xarxa. A partir d'ara ens referirem a ells, generalment, com a intrusos.

Les seves motivacions són diverses, ho són tant que podriem dir que cada *black hat* en té la seva pròpia. No ens queda res més que "conèixer el nostre enemic"[1].

2.2 Millorar la seguretat

Un cop ens hem adonat del perill que suposen els usuaris externs a la nostra xarxa, ens podem preguntar que hem de fer per a poder evitar les intrusions. Un ordinador segur, és aquell que està apagat: un cop encés i connectat a la xarxa és susceptible de tota mena d'atacs, tant d'usuaris, com de virus i programari maliciós. De totes maneres existeixen sistemes per augmentar la seguretat dels ordinadors i intrínsecament de la xarxa mateixa.

Un primer remei, podria ser habilitar un tallafocs o *firewall* a la nostra xarxa, de manera que totes les connexions de qualsevol màquina de la nostra xarxa amb una màquina d'internet passi obligatòriament per el *firewall*. Al *firewall* li aplicarem unes regles per a què accepti i rebutgi paquets tant de la xarxa local com d'internet. Per exemple podem partir de una política de "DROP ALL" amb la que aïllarem la xarxa local de qualsevol connexió tant amb el *firewall* com amb internet. A mesura que apliquem regles per deixar passar paquets començarem a exposar els equips de la xarxa al perill. A més el *firewall* pot ser objecte també d'alguna intrusió.

Una altra solució bona, pot ser la d'instal·lar un sistema de monitorització del tràfic entrant i sortint de la xarxa. D'això se n'ocupen els NIDS (*Network Intrusion Detection System*). Més endavant veurem els avantatges i inconvenients que ens poden comportar aquests tipus de sistemes.

[1] Referència a *Know your enemy* "motto" del Honeynet Project

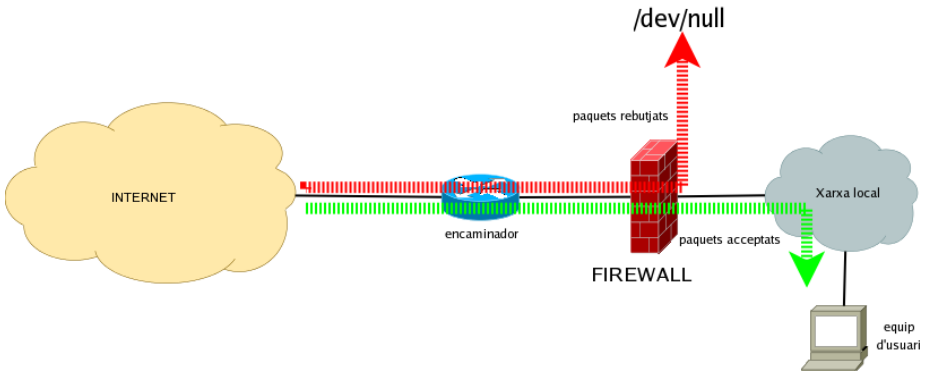


Figura 2.1: Exemple de Xarxa amb *Firewall*

També podem activar SIV (*System Intgrity Verifier*) o LFM (*Log File Monitor*), però més que a mode de sistema, són eïnes per proporcionar certa seguretat en el sistema i que de moment no estan suficientment desenvolupats com per a poder construir un sistema de defensa d'intrusos. Els SIV monitoritzen els fitxers del sistema per què al ser modificats no deixin una *backdoor*[2]. Aquestes eines també controlen quan un usuari obté permisos de superusuari. Els LFM controlen, mitjançant els fitxers de log que deixen altres programes, conductes perilloses per al sistema. Tanmateix és difícil discernir els atacs mitjançant aquesta política.

Finalment, trobem els Honeypots i les Honeynets, altrament anomenats *Deception Systems* que ens serviran per atraure l'atenció dels intrusos a la xarxa, analitzar el seu comportament i poder així millorar la seguretat. Tot seguit veurem el potencial real d'un Honeypot.

2.2.1 La política de l'engany

Un Honeypot és bàsicament un esquer dins la nostra xarxa que utilitzarem per distraure i captar l'atenció de tot aquell que intenti atacar-la. Estan dissenyats per ser atacats i vulnerats amb certa facilitat, imitant i simulant màquines i serveis oferts dins una xarxa determinada.

[2]Traducció de l'anglès: "Porta del darrera" que significa forat de seguretat

Per tant, un Honeypot ha d'oferir certa interactivitat amb l'atacant. Així podrem diferenciar els Honeypot entre *high-interaction* i *low-interaction*. Per exemple, un tipus de Honeypot com pot ser LaBrea:Trapit, denominat com "*sticky honeypot*" (que només agafa adreces IP no utilitzades i crea màquines virtuals per respondre intents de connexió), seria considerat *low-interaction*; mentre que un Honeypot que permeti una intrusió total dins de l'ordinador, que ofereixi serveis com accés a bases de dades, servidors HTTP i tota mena d'informació seria considerat un *high-interaction* Honeypot. Obviament, tota la informació oferida seria fictícia i no permetrem que l'atacant accedeixi a la nostra xarxa "real" o "productiva", del contrari seria contraproduent i altament perillós.

Segons la interactivitat oferta i les característiques de la xarxa, un Honeypot ha de presentar les funcions presentades en la taula 2.2 extret de [16]:

Haurien de...	**No haurien d'oferir**
Proporcionar informació dels atacs en procés	Accés a dades reals
Deixar l'atacant fora de la xarxa de producció i aïllar-lo	Controls administratius a la xarxa real
Proporcionar monitorització a temps real de l'atac	Donar d'alta usuaris
Proporcionar informació de com l'atacant ha entrat	Controlar el tràfic de xarxa
Deixar proves que l'atacant ha entrat per perseguir-lo	Controlar dispositius de la xarxa real

Taula 2.1: Taula de propietats d'un Honeypot

També diferenciem entre Honeypots físics i virtuals:

Honeypot físic és aquell que és implementat en una màquina amb la seva pròpia adreça IP i tots els seus dispositius propis, és a dir, un ordinador ficat com a trampa que pugui ser vulnerat. Més concretament, un exemple podria ser un ordinador amb un servidor HTTP i una sèrie de serveis ficticis. Normalment els Honeypot físics són considerats d'interacció alta. Tanmateix, només tindrà una adreça IP i per tant pot ser que l'atacant passi per alt molts cops aquest Honeypot.

Honeypot virtual Capaç d'apoderar-se de les adreces IP no utilitzades i simular amb elles altres ordinadors. D'aquesta manera ens assegurariem una mica més que l'atacant piqui l'esquer.

El Honeypot és transparent al funcionament habitual de la xarxa, ningú ha d'adonar-se que el Honeypot ha estat posat en marxa, per tant podem seguir amb l'arquitectura que teniem. A més forçosament hem d'intentar camuflar el Honeypot dins la xarxa, oferint serveis propis de la màquina que simulem i de la xarxa que tenim entre mans. Un mal exemple del primer requisit seria el descrit a [7] per Kecia Gubbels.

```
# Example of some simple host templates and their bindings
...
...
annotate ‘‘Windows NT 4 SP3’’ fragment old
create template4
set template4 personality ‘‘Windows NT 4 SP3’’
add template4 tcp port 25 ‘‘sh scripts/smto.sh’’
add template4 tcp port 21 ‘‘sh scripts/ftp.sh $ipsrc $dport’’
set template4 default tcp action reset
bind 100.x.x.102 template4
```

Figura 2.2: Part del fitxer de configuració del Honeyd

En la figura 2.2 podem apreciar, com fem que un servidor funcionant

sobre Windows NT 4.0 simuli un servei de correu SMTP. Quan intentem accedir al port d'aquest servei, trobem un problema:

```
[root@localhost root]# telnet 100.x.x.103 25
Trying 100.x.x.103...
Connected to 100.x.x.103.
Escape character is '^]'.
220 localhost.localdomain.localdomain ESMTP Sendmail
8.12.2/8.12.2/SuSE Linux 0.6}
```

Figura 2.3: Exemple de Telnet al port 25 de la IP 100.x.x.103

L'atacant s'adonaria que un servei de correu que està funcionant amb Linux està sent oferit per un sistema operatiu Windows NT 4.0 (figura 2.3). Sense cap mena de dubte això ens delataria, l'atacant s'adonaria que alguna cosa succeeix i el Honeypot quedaria descobert, cosa que hem d'intentar evitar. També aixecariem suspicàcies entre els atacants si simulessim màquines que no s'utilitzen a la xarxa productiva, per tant hem d'intentar adaptar i camuflar una mica el nostre Honeypot per a què sigui un bon esquer.

Els Honeypot es poden instal·lar en qualsevol sistema operatiu i la majoria són desenvolupats en comunitats i en programari lliure així com les eines utilitzades per a la monitorització dels atacs i la resta d'aplicacions. A continuació fem referència a alguns dels Honeypot i eines més utilitzades:

- **LaBrea:Trapit**: Un dels primers Honeypots implementats. Es tracta d'un Honeypot d'interactivitat baixa basat en sistema Unix. Es denomina *"sticky honeypot"* ja que moltes vegades contesta als intents de connexió de manera que es quedin "atrapats". LaBrea agafa les IP no-utilitzades de la xarxa local i en elles crea màquines virtuals que contesten als intents de connexió per part de l'atacant. Tant el software com els autors han estat guardonats amb diferents premis gràcies a la seva gran tasca. Ara mateix el software no està disponible per descarregar des de la pàgina principal dels autors (www.hackbusters.com)

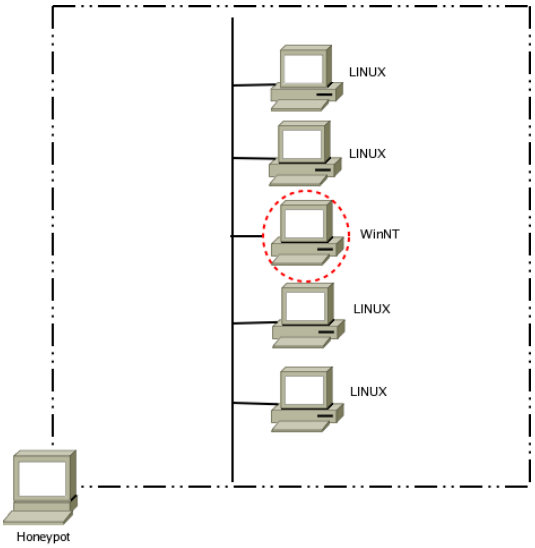


Figura 2.4: Un servidor Windows NT entre molts Linux podria aixecar sospites

degut a canvis en les lleis de l'estat d'Illinois, USA, on s'hostatja el servidor web.

- **Honeyd**: Un dels Honeypots més utilitzat, desenvolupat en OpenSource per a sistemes Unix. Es tracta d'una eina més potent que LaBrea, més fàcil d'instal·lar i configurar. Permet la creació de màquines virtuals per a les IP no utilitzades de la xarxa local. És un *low-interaction* Honeypot, utilitzat tant per a la detecció d'atacs com per activitat desautoritzada. Ha estat desenvolupat i mantingut per Niels Provos de la Universitat de Michigan (*www.honeyd.org*).

- **Nephentes**: És també un Honeypot d'interacció baixa com pot ser-ho Honeyd. Nephentes simula debilitats en el sistema per atraure atacs i reportar logs en aquest context. És força popular per la seva fàcil configuració i intal·lació per a sistemes Gentoo, Debian, OpenBSD i FreeBSD.

- **HoneyBOT**: Es tracta d'un Honeypot d'interacció mitja, desenvolu-

pat per a sistemes MS-Windows per l'empresa Atomic Software Solutions *(http://www.atomicsoftware solutions.com/honeybot.php)*. És un exemple de la versatilitat dels Honeypot quant a sistemes operatius i capacitat d'interacció. El HoneyBOT funciona obrint 1000 sockets udp i tcp simulant serveis vulnerables. Com tots els Honeypot virtuals, escolta aquests ports i reporta logs per a què siguin estudiats posteriorment.

I un cop tenim instal·lat el nostre Honeypot? doncs el podem utilitzar per una infinitat de propòsits. Tots els Honeypots comparteixen la mateixa filosofia: la filosofia de la *deception* o engany. Els fins per al que l'utilitzem generalment seran per a la seguretat, però dins d'aquest sac podem utilitzarlo per la prevenció, detecció o la informació. Segons per a què l'utilitzem podriem distingir entre:

- Honeypots productius: instal·lats en xarxes productives per a captar informació limitada en empreses i corporacions.
- Honeypots de recerca: segueixen polítiques més complexes i recullen molta més informació i molt més completa, utilitzats en universitats, zones militars i governamentals.
- Honeynets: podem considerar una Honeynet com un honeypot de interactivitat alta dedicat a la recerca, ja que el que es pretén amb aquestes eines es crear una xarxa local amb Honeypots i ficar-la darrera d'un sistema de control com pot ser un Firewall. L'objectiu és el mateix, monitoritzar les accions dels intrusos per veure com actuen.

2.2.2 Evolució dels Honeypot

La idea bàsica de Honeypot no és gens innovadora, en altres àmbits fora de la informàtica es fa servir la política de l'engany. El primers textos que contenen la idea de Honeypot són *"The Cuckoo's Egg"* de Clifford Stoll i *"An Evening with Berferd"* per Bill Cheswick publicats abans de l'any 1991.

En ambdós textos s'expliquen encontres amb intrusos i la manera com se'ls combat.

No va ser fins el 1997 quan Fred Cohen publica *The Deception Toolkit* (DTK) un conjunt de scripts fets amb el llenguatge de programació PERL dissenyats sota el sistema operatiu UNIX. Aquest scripts emulen una sèrie de serveis que venen a ser la base ideològica dels Honeypots.

L'anomenat DTK va inspirar la empresa Cybercorp i un any després van treure el primer Honeypot comercial anomenat *CyberCorp Sting* que funciona sobre NT. El mateix any, apareix *NetFacade*: un altre Honeypot comercial més elaborat que el CyberCorp Sting que podia arribar a simular tota una classe C sencera i fins a 7 sistemes operatius. Aquell mateix any apareix un altre Honeypot lliure, basat en Windows, anomenat *Back Officer Friendly* que va apropar una mica més el concepte de Honeypot a la gent.

L'any 1999, Lance Spitzner decideix formar el Honeynet Project, una comunitat de voluntaris sense ànim de lucre dedicats a millorar la seguretat a Internet. Tots els seus treballs són publicats i els seus ideals estan fortament lligats als del Software Lliure. Segons ells mateixos, intenten assolir aquests objectius per aquestes tres vies:

Alerta Augmenten l'alerta dels perills a Internet. Mitjançant les publicacions *Know Your Enemy* divulguen la suficient informació per a què els usuaris i les corporacions siguin conscients dels perills que s'exposen en connectar-se a Internet i a la vegada siguin capaços de protegir-se d'aquests perills.

Informació Un cop som conscients dels perills, el Honeynet Project proporciona informació dels atacants, així com els seus motius, les seves eines i els seus objectius per a què cada usuari o administrador pugui ajustar millor la seguretat en la seva xarxa. Podem aconseguir aquesta informació amb els *Scan of the Month challenges* i els *Know Your Enemy whitepapers*. Els *Scan of the Month challenges* són els resultats de la feina feta per els Honeypots i Honeynets dels col·laboradors del Honeynet Project.

Eïnes Per tot aquell que vulgui seguir augmentant la seguretat, es proporcionen eines desenvolupades per el Honeynet Project.

A partir de la formació del Honeynet Project la majoria de les innovacions en aquest camp han anat sortint d'aquest entorn. L'any 2002 es va formar la Honeynet Research Aliance que va agrupar diferents associacions d'arreu del món per a col·laborar en la mateixa direcció quant a Honeypots es refereix. Han sortit diverses eines relacionades amb els Honeypots des d'aquesta aliança com són: Inline-Snort, Sebek i avançats sistemes de Honeynets virtuals.

2.2.3 NIDS

Anteriorment s'havia parlat dels Sistemes de Detecció d'Intrusions com un sistema de seguretat equivalent als Honeypot i els *Deception Systems*, però en realitat són poc efectius com a sistema. Encara no tenen una maduresa suficient com a sistemes, i entre el diferent programari que podem trobar dins de la mateixa etiqueta hi ha diferents comportaments davant dels atacs. Se'n fa una referència profunda i acurada en [5]. Un altre gran inconvenient en els NIDS rau en la dificultat que suposa determinar si s'està produint un atac o una intrusió mirant el tràfic entrant i sortint de la xarxa. Moltes vegades hi ha alarmes produïdes simplement per paquets erronis. A més, un intrús amb suficients coneixements d'aquests sistemes podria passar desapercebut sense que cap senyal d'alarma ens avisi. Tot això a més de l'ús de codificacions i canals xifrats per a les comunicacions dificulten encara més el treball dels NIDS.

La diferència entre un Honeypot i un NIDS, pot semblar ínfima en un primer moment, però en el moment que s'analitzen ambdos sistemes es veu clarament les diferències que els caracteritzen (figura 2.2).

Malgrat les diferències exposades, els Honeypot i els NIDS no són sistemes excloents un de l'altre, és a dir, podem utilitzar els dos per augmentar la seguretat. De tal manera que els NIDS es converteixen en una eina per monitoritzar el tràfic o per analitzar en cert moment el que passa en el Honeypot

Honeypot	NIDS
Alarma quan el Honeypot ha estat compromés	Alarma quan troba paquets estranys (comporta falses alarmes)
Està aïllat de la xarxa productiva	La xarxa productiva es exposada
Una intrusió compromet el Honeypot	Una intrusió compromet la xarxa real
Difícilment són descoberts	Poden no veure un atac

Taula 2.2: Comparació entre un Honeypot i els NIDS

(es veurà en la part de la implementació com s'utilitzen aquestes eines). El NIDS més famós i utilitzat és Snort. És un OpenSource *Network Intrusion Detection System* capaç d'oferir anàlisi de tràfic en tems real i de reportar logs dels paquets de les xarxes IP. Pot fer anàlisi de protocols, detectar atacs i proves, com overflows, scanneig de ports, atacs CGI (*Common Gateway Interface*), porves SMB (*Server Message Block*) i molt més.

2.2.4 Avantatges contra vulnerabilitat i perills

L'avantatge més gran que ens proporcionen els Honeypot és la seva robustesa com a sistema envers la simplicitat del concepte. També ens permetrà obtenir noves eines i tàctiques utilitzades per els atacants en l'assalt a la nostra màquina anomenades "*zero-days*". La seva lleugeresa fa que amb els mínims recursos es puguin construir grans plataformes de defensa contra atacs a gran escala. Amb un ordenador poc potent podem simular una classe B sencera. A més també hem de tenir en compte la escalabilitat a ipv6, ja que tot i que s'utilitzi xifrat sobre ipv6 el Honeypot actuarà igual com si ho fes amb atacs ipv4.

Però no ens enganyarem, el Honeypot també pot ser vulnerat, no com volem nosaltres controlant tot el que fa l'intrús, sinó perdent el control sobre el Honeypot i desprotegint totalment la xarxa productiva. No només

els intrusos cometen errors, nosaltres també podem passar per alt alguna configuració o quelcom que ens porti a fracàs del sistema.

El primer pas per a què el nostre Honeypot no sigui vulnerat, és que no sigui descobert. Aquest assumpte es tractarà amb més rigorositat en el tema de detecció de Honeypots en el proper capítol, tanmateix cal fer un breu apunt ara. Una raó per la qual el Honeypot pot ser descobert, són les inconsistències. Es tracta de trobar proves de què alguna cosa no va com hauria d'anar per culpa del Honeypot. Una altra possibilitat podria ser trobar traces de programes utilitzats generalment per Honeypots, com pot ser el Vmware. I per últim, podem veure'ns obligats a haver de dir que hi ha un Honeypot en marxa per aspectes legals, llavors també seriem descoberts amb conseqüències imprevisibles.

Com qualsevol altra tecnologia, es pot observar que els Honeypots tenen debilitats. És per això que s'han de fer servir els honeypot amb les eines actuals amb les que es defensen els sistemes. Certament, el honeypot només ens dóna una visió limitada d'allò que succeeix a la xarxa, i a menys que l'atacant passi pel Honeypot no sabrem que passa a les màquines veïnes.

2.3 *Data Forensics*

La traducció al català de l'anomenada *data forensics* és Informàtica forense. Es tracta d'una nova disciplina dins del món de les comunicacions que intenta trobar les proves suficients dins d'un sistema informàtic per a portar-les als tribunals com a evidència d'un delicte. Nosaltres farem el mateix sobre el nostre Honeypot o més ben dit, sobre el nostre *poisoned Honeypot*. Ens referirem al Honeypot que ha estat vulnerat com verinós o enverinat, ja que pot ser un focus de virus, trojans i eines que el *black hat* ha utilitzat per entrar en el Honeypot.

És molt important mantenir tota la informació que ha deixat l'intrús en perfecte estat, és a dir, no modificar ni un bit del que hi ha al disc intentant mantenir fins i tot la memòria volàtil. Ens serà de gran utilitat en cas què

```
Users (authorized or unauthorized) have no explicit or
implicit expectation of privacy.  Any or all uses of this
system may be intercepted, monitored, recorded, copied,
audited, inspected, and disclosed to authorized site,
and law enforcement personnel, as well as to authorized
officials of other agencies, both domestic and foreign.
By using this system, the user consents to such
interception, monitoring, recording, copying, auditing,
inspection, and disclosure at the discretion of authorized
site.

Unauthorized or improper use of this system may result in
administrative disciplinary action and civil and criminal
penalties.  By continuing to use this system you indicate
your awareness of and consent to these terms and conditions
of use.  LOG OFF IMMEDIATELY if you do not agree to the
conditions stated in this warning.
```

Figura 2.5: Un banner que advereteix de la possible monitoritzacio del que fa l'intrús

vulguem presentar la informació recopilada com a prova davant d'un tribunal, fent tot el posible per assegurar que el que presentem no ha estat manipulat. Podem aconseguir-ho fent còpies des de ssh o des d'un live-cd i fent *MD5 sum* de tot allò que tenim.

Un cop arribats aquí comença la feina de debò. En el cas que tinguem un *low-interaction* Honeypot, la feina se'ns simplifica molt. Només haurem de mirar els logs que ha reportat el Honeypot referent als intents d'accés als ports, a partir d'aquí podrem intentar buscar un patró d'atac recopilant poc a poc informació.

La cosa se'ns complica a mesura que la interactivitat puja, podríem dir que a partir d'una interactivitat mitja-alta, on l'atacant haurà entrat real-

ment en el sistema, haurà instal·lat *rootkits* i altre programari i en definitiva haurà fet més mal, haurem d'anar poc a poc seguint els passos que ha donat: començant per on ha entrat, seguint per el que ha fet i intentant parar-lo abans que intenti atacar altres ordinadors des del nostre, d'una altra manera ens podríem ficar en un bon embolic.

Un cop tinguem tota la informació copiada, verificant la seva autenticitat, copiarem un altre cop tota la informació en un disc dur per a procedir a la seva anàlisi. Aïllarem l'ordinador en una xarxa controlada, per els perill que pot comportar. Hem d'anar amb peus de plom ja que el Honeypot segurament estarà ple de *malware* i podríem executar algun d'aquest programes, amb tot el que això suposa. Ara, el nostre objectiu és repassar tots els fitxer modificats, revisar els log i mirar de descobrir que és allò que ha instal·lat l'intrús, com ho ha fet i perquè. Realment es tracta d'una feina molt feixuga. Una hora d'atac pot comportar quaranta hores d'anàlisi.

2.4 Legalitat

En termes de seguretat els Honeypot ens aporten una gran quantitat d'informació que pot solucionar certs problemes a la xarxa on s'implementen. Però, són els honeypot legals? Hi ha una sèrie de variables que fan que aquesta pregunta sigui difícil de contestar. Tot depèn del país on resideixes, la informació que emmagatzemes i que fas amb ella i la aplicació que li dones a la tecnologia. Ni tan sols en els Estats Units (on s'ha discutit molt la qüestió) està clar si l'ús dels Honeypot és legal, ja que depèn de les lleis de cada estat.

Abans i tot de saber si l'estat on resideixes pena l'ús d'aquestes tecnologies, podriem estar violant les polítiques de privacitat de la organització on s'implanten els honeypots. Per tant hi ha moltes preguntes a fer-nos abans de començar a ficar esquers a la xarxa.

Poden els honeypot invair àrees protegides dels usuaris fins i tot si aquests han entrat il·lícitament? Quanta informació podem agafar de l'intrús que entra al nostre honeypot? Què podem arribar a fer amb ella? Què passa si

algú de fora entra i ens omple de pornografia infantil o arxius de pirateria?

Intentarem abordar l'assumte des de tres punts de vista diferents:

Privacitat La privacitat se centra en la confidencialitat de la informació. Està clar que els honeypot d'interactivitat baixa no es veuran afectats per la violació de privacitat, però els d'interactivitat alta poden aportar molta informació de tot aquell qui ha passejat per dins, com pot ser: converses IRC, correus electrònics, conexions a pàgines segures, emmagatzemament de contrasenyes i claus privades, etc. I no només informació de l'intrús sinò també informacio de terceres persones amb les que interactua.

Inducció a la comisió d'un delicte L'ús dels honeypot per a l'administració de la seguretat es pot veure com un cas d'inducció a un delicte, ja que estem provocant a algú per a què delinqueixi, cosa que és punible a menys que siguis una força de seguretat de l'administració. Es podria fer una comparació amb deixar les finestres de casa teva obertes i esperar a dins per estomacar al lladre.

Responsabilitat Si deixem que algú entri en una màquina nostra per a què faci el que vulgui podem tenir certs problemes si l'intrús ens utilitza per a delinquir, ja sigui fent servir el honeypot per emmagatzemar pornografia infantil, fitxers pirata, atacar altres màquines, etc., segurament podriem al·legar que hem estat víctimes d'un delicte telemàtic, però ens veuriem ficats dins d'una problemàtica força gran.

2.5 Èxits dels Honeypots i Honeynets

Durant els anys 2000 i 2001 hi va haver un gran augment d'infeccions tant en UNIX com en Windows de cucs informàtics [1]. La gran habilitat que tenen aquests programes per distribuir-se automàticament per tota la xarxa

[1] Un cuc és un virus informàtic o programa autoreplicant que no altera els fitxers sinò que resideix en la memoria i es duplica a ell mateix, contagiant així altres equips

els convertia en un gran perill. A més, el repte de tot administrador era poder aconseguir una copia del cuc per entendre com funciona, analitzar-lo i finalment protegir els sistemes d'aquest cuc. Aquest procés era inviable en casos com el *CodeRed*, ja que buscar entre la informació d'una xarxa en producció per trobar un cuc és com buscar una agulla en un paller, a més el *CodeRed* només residia en la memòria del sistema. Aquest exemple va servir per a demostrar a la comunitat informàtica la vàlua dels Honeypot.

Un bon exemple és la captura del cuc *Leaves*. Al juny del 2001 es va detectar un augment considerable d'escanejos del trojà *Sub7*. El que feia bàsicament aquest trojà es permetre l'accés remot al sistema per apoderar-se d'aquest mateix mitjançant un software especial. Un equip de la companyia Incidents.org van desenvolupar un sistema per trobar la rao d'aquest comportament.

Johannes Ullrich del SANS Institute va desenvolupar un honeypot que emulava un sistema Windows infectat amb el *Sub7*. En pocs minuts es va rebre un atac que va permetre la captura de tota la informació necessària per a l'anàlisi. Es va descobrir un cuc que simulava ser un client de *Sub7* i d'aquesta manera infectar el sistema. El que aconseguia l'atacant era infectar el sistema amb el seu cuc sense haver de desenvolupar una manera d'entrar a l'equip, ja que aprofitava que el sistema ja estava infectat amb el trojà *Sub7*. A partir de llavors moltes organitzacions van començar a utilitzar Honeypots per a la captura de cucs i un anàlisi posterior.

La primera captura d'un atac desconegut va ser al Gener del 2002 quan un honeypot Solaris va capturar un atac dtspcd, un atac nou mai vist anteriorment. Uns mesos abans el CERT va divulgar un avís per el CDE (*Subprocess Control Service*), més concretament per el dimoni dtspcd. Es va informar de què aquest servei era vulnerable, i que teòricament es podia guanyar el control de la màquina remotament. A pesar d'això no constava cap atac aprofitant aquesta debilitat i es creia que ningú el feia servir. El honeypot va demostrar que la comunitat de *black hats* atacaven mitjançant aquesta debilitat.

Des de l'aparició d'aquesta tecnologia s'ha fet patent que els Honeypot poden ser la nostra millor arma en els següents camps:

- Detecció i prevenció d'atacs automatitzats
- Detecció i prevenció d'atacs humans
- Mètodes de detecció precisa
- Eina Ciber-Forense

Capítol 3

Funcionament d'un Honeypot

En aquest capítol intentarem per una banda entendre com funciona realment un honeypot i com es pot implementar per a què sigui una eina útil. Per altra banda intentarem introduir solucions reals que més tard adaptarem a la nostra xarxa productiva.

Hem de dir que hi ha moltes solucions per a ficar en funcionament un honeypot:

- implementacions comercials que ens facilitaran la nostra feina ja que només haurem d'instal·lar el producte i esperar resultats.

- solucions lliures, un software desenvolupat per una comunitat que ens permetrà diferents configuracions i modificacions per tal d'adaptar-lo a les nostres necessitats.

- el nostre propi honeypot, adaptant-lo completament al que necessitem.

En el present document, s'ha tractat el tema en la vessant més desenvolupadora, és per això que no té sentit aportar sol·lucions comercials ni de codi tancat, per tant, tot el software utilitzat té les seves fonts accessibles. Tot el software utilitzat és executat en un entorn UNIX, aprofitant així la seguretat i les eines que ens pot aportar, més concretament s'ha utilitzat GNU/Linux amb una distribució Fedora Core 5.

3.1 Honeyd

El primer honeypot utilitzat, molt estés dins la comunitat d'administradors gràcies a la seva facilitat de funcionament i implementació. Es tracta d'un petit dimoni que crea hosts virtuals dins d'una xarxa. A cada host se li pot assignar una personalitat diferent, de manera que en una mateixa màquina puguem crear diferents hosts virtuals que responguin amb un funcionament diferent. Honeyd és software de codi obert sota la Llicència Pública General GNU. Actualment està mantingut i desenvolupat per Niels Provos en la versió Honeyd 1.5b publicada el 19-08-2006.

El disseny i la implementació del Honeyd es basa en limitar la interactivitat només a nivell de xarxa, és a dir, que no se simulen tots els aspectes del sistema operatiu, sinó que només se simula la pila de xarxa. D'aquesta manera aconseguim que si l'atacant guanya accés, no comprometi el sistema operatiu. Per aquesta raó el Honeyd és considerat un honeypot de'interacció baixa que simula serveis TCP i UDP. També entén i respon a missatges ICMP.

Honeyd està dissenyat per respondre a paquets amb la direcció IP de destí que correspongui a una de les adreces que el Honeyd simuli. Això es pot fer de diverses maneres que seran descrites més endavant. A més ha de permetre enganyar l'adversari de manera que mai sàpiga que es tracta d'un honeypot. Podem simular equipament de xarxa de nivell 3, fet que es pot aprofitar per crear honeynets amb routers virtuals i fingint amples de banda i pèrdua de paquets.

L'arquitectura del Honeyd(figura 5.2) consisteix en diferents components, els més importants són un tractador de paquets, els manipuladors de protocols i un motor de personalitats.

Els paquets entrants són processats per el tractador de paquets, es comprova el *checksum* i la longitud de paquet, llavors es mira si és un paquet TCP, UDP o ICMP. Qualsevol altre paquet d'un altre protocol es fa un *log* i es descarta. Per als paquets ICMP només es respon a les peticions d'*echo* i a la resta de peticions es processen missatges de *destination unreacheable*.

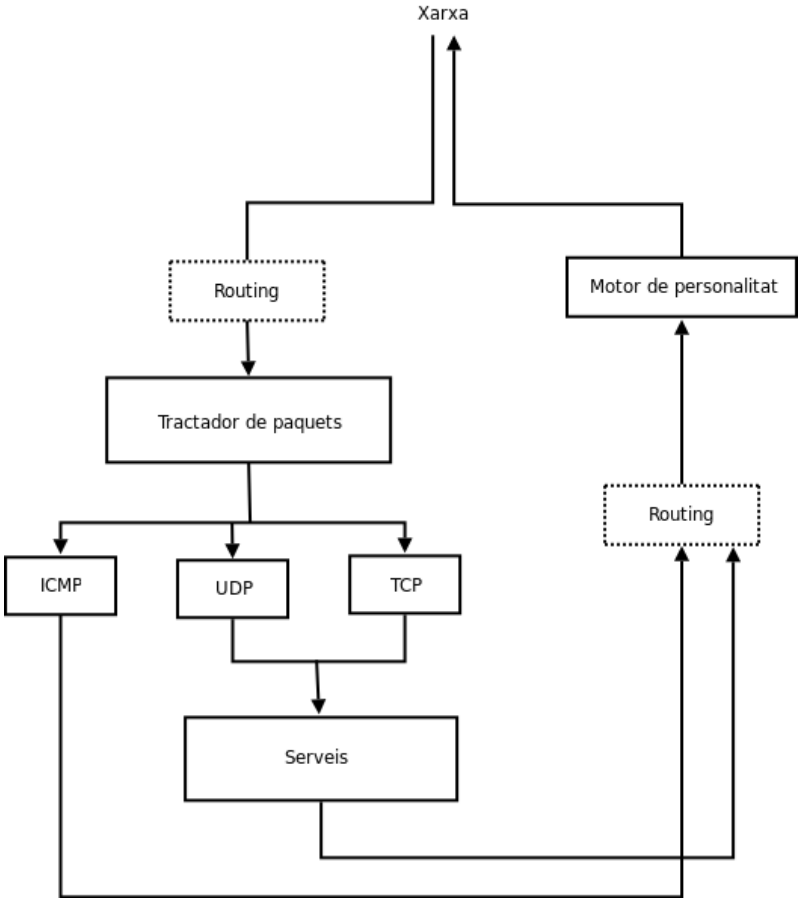


Figura 3.1: Arquitectura de Honeyd

Abans de processar el paquet es mira si per aquella adreça IP hi ha una configuració definida, sinó s'utilitza la configuraciò per defecte. Per paquets TCP i UDP, es mira si es tracta d'una connexió nova. Si es tracta d'una connexió nova es crea un nou servei. Els serveis són aplicacions externes al dimoni però residents en el mateix sistema, que reben i envien informació al Honeyd. Abans que el paquet es torni a enviar a la xarxa, és tractat per el motor de personalitat, que li ajusta el contingut per a què sembli que aquest ha estat originat per la pila de xarxa del sistema operatiu desitjat. Honeyd utilitza la mateixa base de dades de signatures que utilitza Nmap[1]

[1]Nmap és un programa OpenSource que serveix per al rastreig de ports. S'utilitza per

per respondre a les proves del sistema operatiu emulat.

3.1.1 Instal·lació i configuració

La instal·lació de Honeyd és força sencilla, només cal resoldre unes depèndencies de les llibreries *libevent*, *libdnet* i *libpcap*.

- *libdnet* proporciona una interactuació simplificada a una sèrie de rutines de baix nivell com: manipulació d'adreces de xarxa, arp cahce del kernel, etc.

- *libpcap* proporciona accés independent de la implementació a la captura de paquets proporcionada per el sistema operatiu.

- *libcap* és la llibreria utilitzada per capturar paquets de la tarja de xarxa.

La configuració de Honeyd requereix un fitxer en el qual s'especificarà el seu comportament, en aquest fitxer es descriuran una sèrie de plantilles amb la seva personalitat, també s'assignarà el comportament que aquestes tindràn. Finalment es lligarà aquesta personalitat a una o diverses adreçes. Tenim un exemple a la figura 3.2.

```
annotate "FreeBSD 2.2.1 - 4.1" fragment old
create template2
set template2 personality "FreeBSD 2.2.1 - 4.1"
add template2 tcp port 23 proxy $ipsrc:23
add template2 tcp port 53 proxy $ipsrc:53
add template2 tcp port 110 "sh scripts/pop3.sh"
set template2 default tcp action reset
bind 100.x.x.101 template2
```

Figura 3.2: Exemple de personalitat FreeBSD.

avaluar la seguretat de sistemes informàtics, així com per descobrir serveis o servidors d'una xarxa informàtica.

La personalitat es extreta de les *fingerprints* del programa Nmap. Aquesta base de dades és d'on el Honeyd treu les signatures per emular les piles IP dels diferents sistemes operatius.

Com ja s'ha dit, el Honeyd treballa creant plantilles que defineixen les característiques del Honeypot: tipus de sistema operatiu, els ports que escolta i com reacciona davant els diferents serveis. Cada plantilla té un nom i es poden crear tantes plantilles com es desitgi. Tot seguit es vinculen les plantilles a les adreces IP que es vulguin simular. També existeix una plantilla anomenada *default* que no pot ser vinculada a cap IP.

Un cop s'ha determinat la personalitat del honeypot, hem de saber el comportament de cada servei, és a dir, com reacciona el honeypot davant la connexió a un port específic. Podem configurar el nostre honeypot per a què reaccioni de les següents maneres per a les connexions als ports UDP i TCP:

Reset: Aquesta opció enviarà un paquet TCP amb el flag *RST* activat o un paquet ICMP *destination unreacheable* per a les connexions UDP i ICMP. Això significarà que el port està tancat.

Open: Això significarà que el honeypot respondrà amb un paquet amb el *flag ACK* activat per a les peticions de connexió TCP. Tanmateix, no hi ha cap servei en marxa.

Block: El honeypot rebutjarà el paquet i ignorarà la connexió tal i com faria un firewall.

Script: Cridarà a un script per interactuar amb l'atacant. Amb aquests scripts s'emularan els serveis. Limitat només al protocol TCP.

Honeyd també permet la utilització de plantilles dinàmiques, que donen l'habilitat de canviar el comportament segons algunes condicions:

source adress segons l'adreça d'origen de la xarxa es determina el comportament que s'utilitzarà.

operating system el sistema operatiu ha de ser trobat per la plantilla per tal de ser activada.

time la plantilla només és utilitzada en un interval de temps. Això permet Honeyd simular equips que són encesos i apagats.

Un exemple de plantilla dinàmica és el següent:

```
dynamic magichost
add magichost use windowsxp if source os = windows
add magichost use linux if source ip = 192.168.0.0/16
add magichost use invisible if time between 12:00am - 5:00am
add magichost otherwise use default
```

Un aspecte molt important, és la manera d'emmagatzemar la informació. Honeyd té diverses formes de capturar l'anomenat *logging*, la més usual de les quals és que el pròpi dimoni salvi cada nou intent de connexió per tots els protocols, indicant la adreça IP origen i el port de connexió.

Per altra banda, si cada servei reporta la informació aconseguida, aquesta és de major rellevància ja que podem veure una mica més d'aprop el que l'atacant intenta fer. Per exemple, si simulem una connexió per ssh podem obtenir el login i el password que l'atacant utilitzarà.

Per últim Honeyd es pot utilitzar juntament amb un NID: a part d'aconseguir més informació i de més qualitat podem aportar una mica més de seguretat al honeypot ja que normalment ens estarem veient les cares amb adversaris maliciosos. Podem aprofitar la documentació facilitada per l'autor que recomana systrace per actuar en conjunció amb el Honeyd.

3.1.2 Blackholing

Blackholing és l'estratègia més senzilla per tal de dirigir el tràfic cap al Honeyd per a fer que aquest respongui. Es tracta d'afegir una ruta d'encaminament al router de la xarxa per a fer que aquest dirigeixi tots els paquets

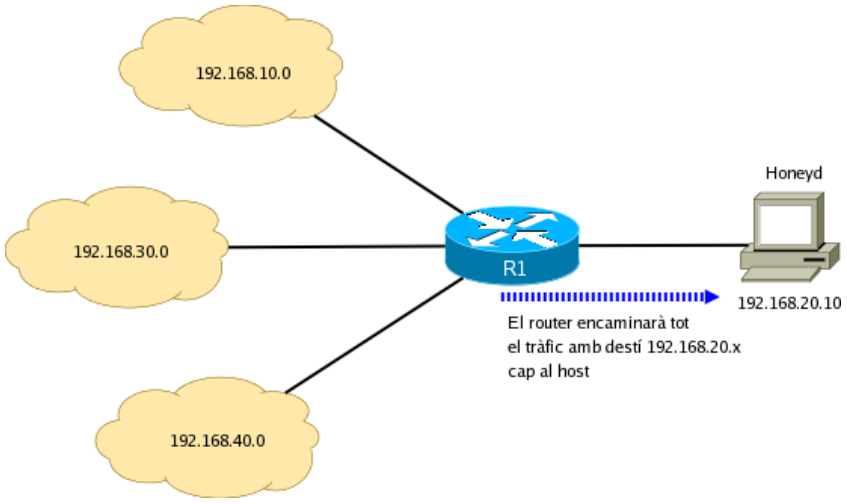


Figura 3.3: Esquema del Blackholing

```
IP ROUTING TABLE
----------------
Group:VLAN
Network         Mask            Gateway         Metric     Id     Protocol
-------------------------------------------------------------------------
10.0.0.0        255.0.0.0       10.50.23.123       1       2:1     NO ROUTING
192.168.10.0    255.255.255.0   192.168.10.1       1       1:1     DIRECT
192.168.20.0    255.255.255.0   192.168.20.10      1      20:1     STATIC
192.168.30.0    255.255.255.0   192.168.30.1       1      30:1     DIRECT
192.168.40.0    255.255.255.0   192.168.40.1       1      40:1     DIRECT
```

Figura 3.4: Taula d'encaminament per a Blackholing

que ens interessen cap a la interfície del honeypot. Així doncs aquest podria emular els host de la xarxa que encaminem cap a ell.

Amb la configuració presentada en la figura 3.3, podem fer que el Honeyd emuli qualsevol màquina de la classe C 192.168.10.0. El cost que tenim és que hem d'afegir rutes d'encaminament als equips de xarxa, i això és sempre un mal de cap per a l'administrador, a més tindrem ocupada tota la classe C que és un preu que moltes vegades no podrem pagar. A més sempre dependrem

de l'encaminament. Així doncs, a nivell 2 la guerra està perduda. És a dir si hem de ficar màquines reals en el mateix medi, les comunicacions no passaran per el router i per tant no s'encaminaran les adreçes "falses" cap al Honeypot. Tot això fa que l'estratègia del Blackholing sigui del tot ineficient.

3.1.3 ARP Spoofing

En primer lloc, farem una breu descripció del que fa el protocol ARP: Es

+	Bits 0 - 7	8 - 15	16 - 31
0	Hardware type (HTYPE)		Protocol type (PTYPE)
32	Hardware length (HLEN)	Protocol length (PLEN)	Operation (OPER)
64	Sender hardware address (SHA)		
?	Sender protocol address (SPA)		
?	Target hardware address (THA)		
?	Target protocol address (TPA)		

Figura 3.5: Estructura d'un paquet ARP

tracta d'enviar un paquet ARP-request a l'adreça broadcast preguntant qui té una certa adreça de nivell 3. Tots els equips reben aquesta trama però només l'equip que té l'adreça requerida la tracta i envia un paquet de resposta ARP-reply. L'equip que ha començat la transmisió emmagatzema l'adreça MAC en la seva taula i comença la transferència de paquets.

Intentarem aprofitar aquest protocol per enganyar l'adversari, responent a totes les peticions de ARP no contestades. És a dir, respondrem per totes aquelles IP no assignades dins de la nostra xarxa, aquesta estratègia és l'anomenat ARP spoofing, que ens permetrà emular equips en aquelles IP inutilitzades, sense haver de configurar res més que el nostre equip, la qual cosa ens proporciona molta més llibertat i un punt de refinament en la eficiència.

Tot i que la idea és força sencilla, necessitem el software adient:
Arpd és un dimoni que respon per les adreces IP que no han estat assignades, tot i que està desenvolupat per a què funcioni sota sistemes UNIX, els kernels de les distribucions més populars no tenen el suport per arpd; com Fedora

Core no el té, hem de recompilar el kernel.

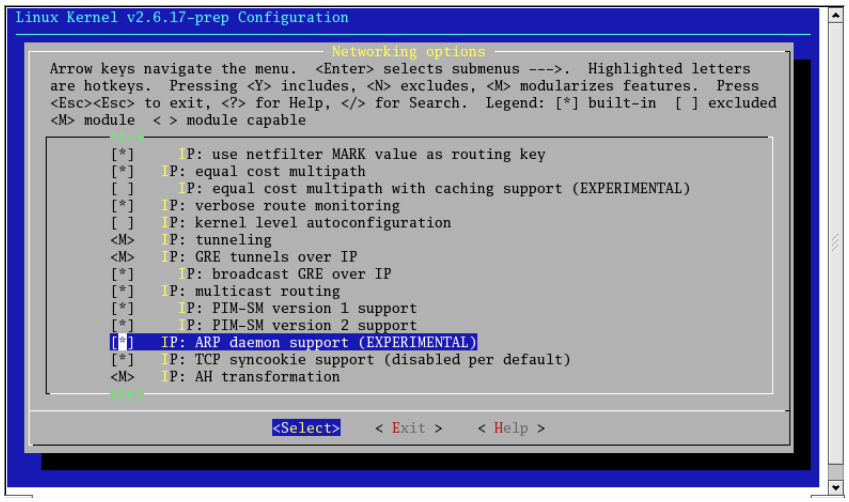


Figura 3.6: Opció del menuconfig del kernel de Fedora Core 5

A part, ens queda instal·lar les llibreries necessàries per a què aquest dimoni pugui funcionar, no és una tasca sencilla, ja que la majoria són desenvolupades per a entorns BSD i s'ha de canviar algunes línies del codi per a què pugui compilar en un entorn GNU/Linux.

A pesar de la dificultat en la instal·lació, els resultats són immillorables, utilitzant arpd amb Honeyd tenim un honeypot virtual força potent. En la figura 3.7 podem veure com arpd respón per certes adreces IP.

```
[root@arale ~]# arpd -d 192.168.30.12
arpd[2549]: listening on eth0: arp and (dst 192.168.30.12)
and not ether src 00:02:b3:b3:5c:8c
arpd[2549]: : no entry for arpd_lookup
arpd[2549]: : who-has arpd_send tell 192.168.30.12
arpd[2549]: : who-has arpd_send tell 192.168.30.12
arpd[2549]: arpd_recv_cb: 192.168.30.12 still discovering (2)
arpd[2549]: arp reply 192.168.30.12 is-at 00:02:b3:b3:5c:8c
arpd[2549]: arp reply 192.168.30.12 is-at 00:02:b3:b3:5c:8c
arpd[2549]: arp reply 192.168.30.12 is-at 00:02:b3:b3:5c:8c
```

```
arpd[2549]: arp reply 192.168.30.12 is-at 00:02:b3:b3:5c:8c
arpd[2549]: arp reply 192.168.30.12 is-at 00:02:b3:b3:5c:8c
arpd[2549]: exiting on signal 2
```

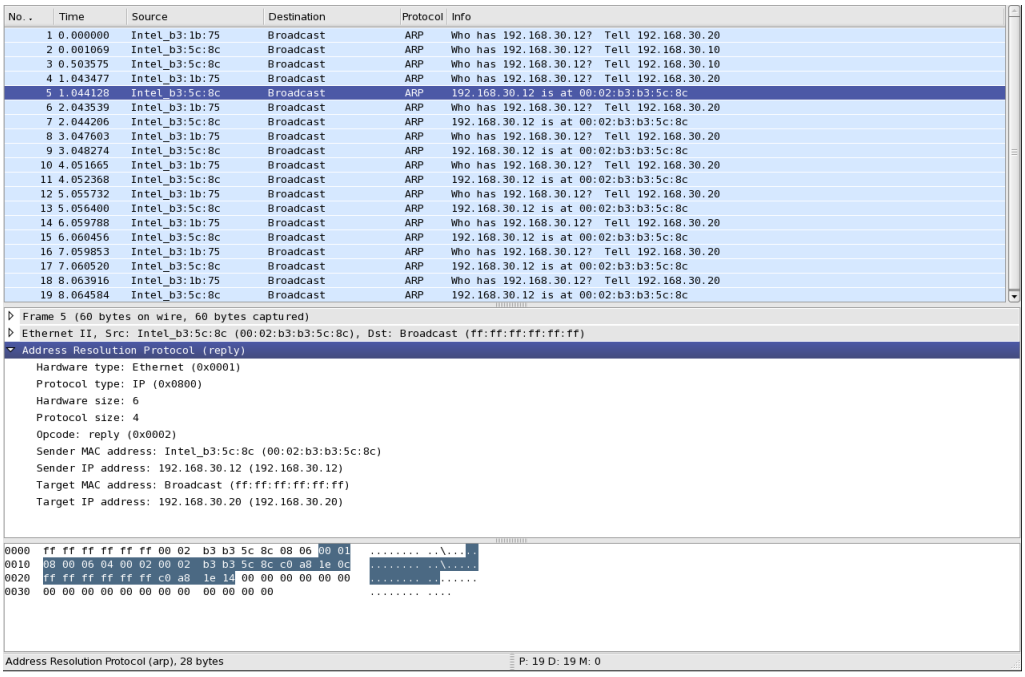


Figura 3.7: Captura del tràfic de la xarxa de l'ARP spoofing

3.1.4 ARP Proxy

L'ARP proxy és una alternativa per l'ARP spoofing. Com l'ARP spoofing, l'objectiu principal és vincular unes adreces MAC a unes adreces IP, el que aconseguiem amb l'arpd era fer-ho d'una manera dinàmica per aquelles adreces IP no utilitzades en un cert moment. En canvi amb proxy ARP es fa d'una manera estàtica. La majoria de sistemes UNIX permeten l'ARP proxy, generalment l'opció està desactivada per defecte.

Per activar-lo:

```
# echo 1 > /proc/sys/net/ipv4/conf/default/proxy_arp
```

Per exemple: imaginem que estem dins d'una classe C amb adreçament 192.168.1.0, reservem 5 adreces que mai seran assignades a cap equip: 192.168.1.200-204. En lloc de monitoritzar la xarxa, podem assignar estàticament aquestes 5 adreces a la nostra MAC (on tindrem el Honeyd). Utilitzarem la comanda `arp -s` que assignarà estàticament la IP a la nostra MAC.

```
arp -s 192.168.1.200 00:02:3F:3A:F5:D1 permanent pub
arp -s 192.168.1.201 00:02:3F:3A:F5:D1 permanent pub
arp -s 192.168.1.202 00:02:3F:3A:F5:D1 permanent pub
arp -s 192.168.1.203 00:02:3F:3A:F5:D1 permanent pub
arp -s 192.168.1.204 00:02:3F:3A:F5:D1 permanent pub
```

Si un atacant envia algun paquet a alguna de les 5 adreces IP, el nostre equip contestarà les peticions ARP per aquestes IP.

3.1.5 Simulant *routing* ample de banda i pèrdua de paquets

Fins ara s'ha pogut veure com Honeyd és un honeypot molt potent quant a la simulació de hosts virtuals, ara veurem unes quantes configuracions una mica més avançades amb les quals arribarem a simular xarxes virtuals senceres, anomenades honeynets que ens poden aportar molta més informació valuosa en un entorn de recerca.

Simulació d'un router

La xarxa que simularem utilitza un adreçament 10.0.1.0/24. Conté dos honeypots i està separada de la LAN per un router Cisco (R1) com mostra la figura 3.8.
Primer crearem la plantilla del router Cisco

```
 #Router Cisco
create router
set router personality "Cisco IOS 11.3 - 12.0(11)"
```

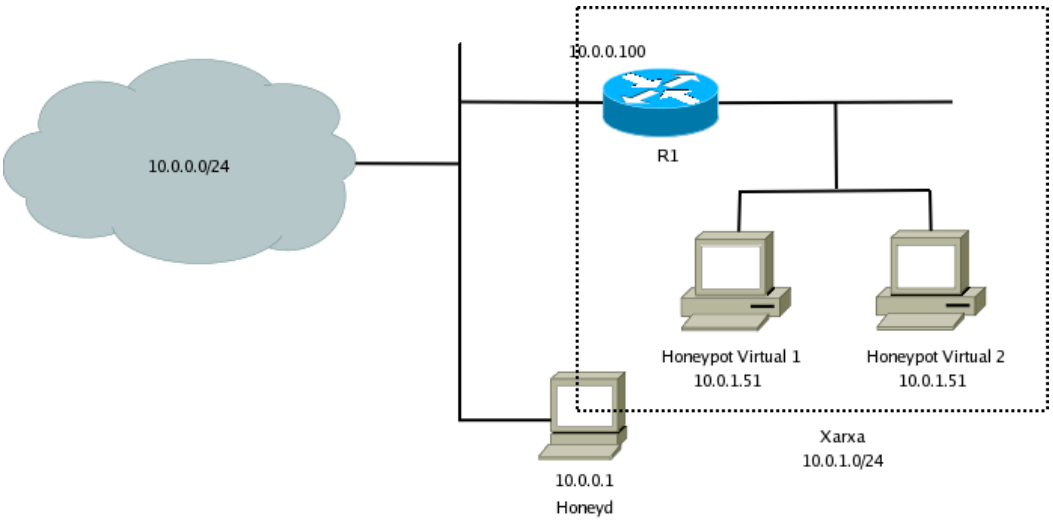


Figura 3.8: Esquema de la xarxa amb un router simulat

```
set router default tcp action reset
set router default udp action reset
add router tcp port 23 "/usr/bin/perl scripts/router-telnet.pl"
set router uptime 1324343
bind 10.0.0.100 router
```

El router R1 és el punt d'entrada a la xarxa virtual de la LAN; la comanda `router entry` especificarà el punt d'entrada.

```
route entry 10.0.0.100 network 10.0.0.0/16
```

És possible tenir múltiples routers d'entrada i que cadascun faci servir diferents rangs de xarxa.

La xarxa 10.0.1.0/24 és directament accessible des del router R1. La comanda `router link` s'utilitza per especificar quina xarxa és directament accessible i no es necessita cap més salt per arribar-hi.

```
route 10.0.0.1000 link 10.0.1.0/24
```

La primera IP és la del router. L'adreça de xarxa especificada després de `link` defineix quina xarxa és accessible. Es poden utilitzar diverses comandes

`link` per definir múltiples xarxes connectades al router.

Per últim, els dos honeypots amb IP 10.0.1.51 i .52 es configuraran vinculant les dues adreces amb dues plantilles.

```
bind 10.0.1.51 freebsd
bind 10.0.1.52 solaris
```

Simulacio de dos routers virtuals

Un cop hem vist com configurar un router virtual, ens proposem afegir una nova porta d'enllaç. A la figura, 3.9 hem afegit un nou router R2 separat de la primera xarxa amb adreçament IP 10.0.1.100. El nou rang d'adreces és 10.1.0.0/16 i hi haurà dos hosts a les adreces 10.1.0.51 i 10.1.0.52.

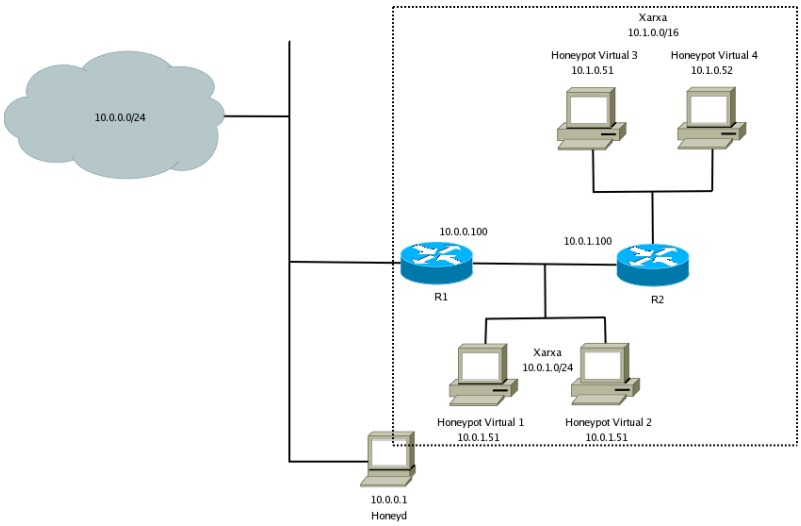


Figura 3.9: Esquema de la xarxa amb dos routers simulats

Primer, afegirem una nova porta d'enllaç (R2) al nostre fitxer de configuració. La comanda serà:

```
route 10.0.0.100 add net 10.1.0.0/16 10.0.1.100
```

Això especifica que el R1 pot accedir la xarxa 10.1.0.0/16 passant per la porta d'enllaç 10.0.1.100 (R2). La primera IP de la línia és la del R1, l'última adreça

IP és la de la nova porta d'enllaç, l'adreça de xarxa és la nova xarxa que es podrà accedir via la nova porta d'enllaç.

Un cop hem afegit el R2, necessitem definir quines IP seràn accessibles directament des del R2. Com ja hem vist abans, utilitzarem la comanda `link`.

```
route 10.0.1.100 link 10.1.0.0/16
```

Només ens queda afegir els dos host virtuals vinculant dues adreces IP amb dues plantilles:

```
bind 10.1.0.51 linux
bind 10.1.0.52 win
```

Configurant latència, pèrdua de paquets i ample de banda

Per a mostrar com emular latència, pèrdua de paquets i ample de banda; afegirem un tercer router a un salt de R2.

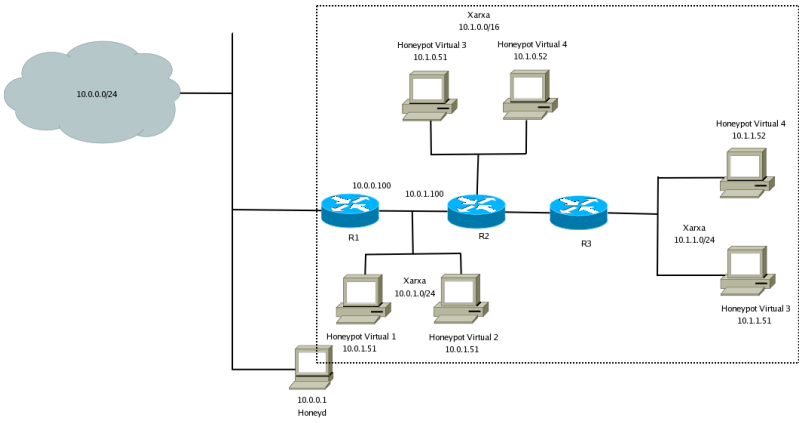


Figura 3.10: Esquema de la xarxa amb tres routers simulats i alguns paràmetres més

```
route 10.0.1.100 add net 10.1.1.0/24 10.1.0.100
latency 50ms loss 0.1 bandwidth 1Mbps
```

```
route 10.1.0.100 link 10.1.1.0/24
bind 10.1.1.51
bind 10.1.1.52
```

El que hem fet amb aquestes línies de configuració ha estat afegir la IP 10.1.0.100 com a porta d'enllaç per a poder arribar a la xarxa 10.1.1.0/24, i a més emular dos host virtuals més. Addicionalment també hem configurat latència, pèrdua de paquets i ample de banda amb les comandes `latency`, `loss` i `bandwidth`.

3.1.6 Anàlisi

Honeyd ens proporciona una bona font d'informació, la qual es obtinguda de dos llocs:

- syslogd
- sistema d'*sniffing*

Així doncs podrem capturar totes les connexions TCP establertes del syslogd així com les peticions ICMP d'echo. Algunes altres connexions també podran ser capturades si s'ha facilitat algun procés de log dins el Honeyd (com pot ser un script).

També podrem fer servir el Honeyd com a sniffer per a capturar l'activitat de l'atacant. Podriem instal·lar un Snort al nostre honeypot per aportar més informació, tot i que ens trobarem amb el problema del xifrat de les dades.

Honeyd tampoc aporta un sistema d'informació per als atacs, això comporta que s'hagi de revisar el log del Honeyd per a mirar si hi ha alguna situació crítica. Això pot convertir-se en un problema si hem de revisar l'activitat de molts honeypots. Tanmateix, existeixen solucions com pot ser Swatch. Swatch revisa automaticament els fitxers de log reportant l'incident de una manera customitzable, per exemple amb un e-mail a l'administrador.

Des d'un punt de vista de seguretat, Honeyd introdueix un risc limitat. Com es pot deduïr, això ve donat per la baixa interactivitat oferta per

Honeyd, amb el qual mai facilitarem interactivitat amb el nostre sistema operatiu.

En resum, Honeyd és una solucio OpenSource d'interactivitat baixa que ens permetra la monitorització de milions de sistemes i asumir moltes personalitats al mateix temps. En la següent taula analitzarem els avantatges i desavantatges:

Avantatges de Honeyd	**Desavantatges de Honeyd**
Pot monitoritzar conexions UDP i TCP	El tractar-se d'una sol·lució d'interactivitat baixa, no pot proporcionar interacció amb el sistema operatiu
Es gratuït i de codi lliure, serà desenvolupat ràpidament amb l'ajut de la comunitat	No proporciona support formal
Pot simular diversos sistemes operatius al nivell de la pila IP i a la capa d'aplicació	No pot capturar massa informació

Taula 3.1: Avantatges i desavantatges de Honeyd

3.2 Nepenthes

En aquesta secció introduirem la plataforma Nepenthes. Intentarem mostrar com el concepte de low-interaction honeypot pot ser emprat per capturar malware. I com a la vegada es pot utilitzar aquesta plataforma per analitzar patrons d'atac.

Nepenthes no es defineix com un honeypot per si mateix, sinó com una plataforma per a afegir mòduls honeypot (anomenats mòduls de vulnerabilitat). D'aquesta manera obtenim un honeypot d'interacció baixa amb molta expresivitat.

La idea principal en la plataforma Nepenthes és l'emulació de serveis de vulnerabilitat, que a diferència dels scripts del honeyd, permet emular protocols més complexos com pot ser l'emulació completa del protocol FTP i per tant podem obtenir molta més informació i molt més valuosa.

3.2.1 Arquitectura

Nepenthes està basat en una estructura molt flexible i modularitzable. El nucli és qui manipula la interfície i qui coordina tots els moduls. Per altra banda, els mòduls són:

- *Vulnerability modules*: emulen les parts vulnerables dels sistemes de xarxa
- *Shellcode parsing modules*: analitzen les dades rebudes per els *vulnerability modules*: extreuen informació sobre el malware.
- *Fetch modules* extreuen informació del codi per tal de baixar-se el malware d'una màquina remota.
- *Submision modules*: s'encarreguen del malware descarregat.
- *Logging modules*: reporten informació de les dades obtingudes i aporten informació dels patrons d'atac.

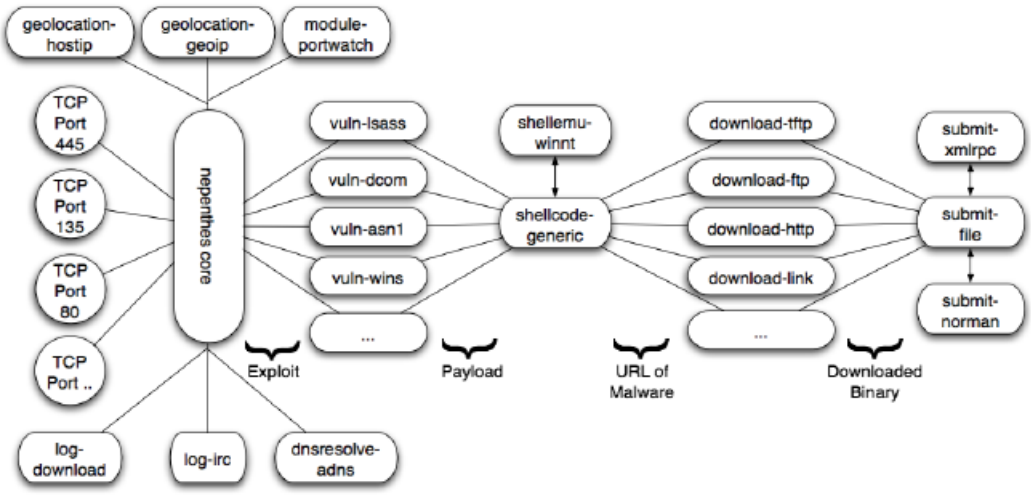


Figura 3.11: Esquema de l'arquitectura de la plataforma Nepenthes

Adicionalment, molts altres components són importants per la funcionalitat i la eficiència de la plataforma nepenthes:

- *shell emulation*, un sistema de fitxers virtuals per cada *shell* emulada.
- *geolocation modules*, un sistema de localització geogràfica.
- *sniffing modules*, per a conèixer més sobre l'activitat en els ports específics i resolució asíncrona de DNS.

En la figura 3.11 s'exposen els mòduls en interacció amb els altres i tot seguit farem una descripció detallada del funcionament de cadascun.

Vulnerability modules

Els *Vulnerability modules* són la clau de la plataforma nepenthes. Són els que activen el mecanisme per recol·lectar el malware. En lloc d'emular un servei sencer, aquest mòduls només emulen una part, ja que per infectar-se de malware autònom emulant les parts necessàries, és suficient. Fins i tot, de vegades, aportant la informació mínima en el moment precís durant el

procés d'atac és suficient com per enganyar al malware. Així doncs un cop obtenim les dades del malware podem passar aquesta informació al següent mòdul.

Shellcode parsing module

Els *Shellcode parsing modules* analitzen les dades aportades per els *vulnerability modules* per poder extraure informació rellevant sobre l'intent d'atac. El seu funcionament és el següent:

- Primer decodifiquen el codi. La majoria dels codis estan codificats amb XOR, d'aquesta manera poden evitar la detecció per part dels NIDS.
- Tot seguit intenta trobar algun patró conegut o una URL.

Els resultats seràn analitzats més endavant i si hi ha suficient informació reconeguda, es passa al següent mòdul.

Fetch modules

Els *Fetch modules* tenen la funció de descarregar els fitxer des d'un lloc remot. Actualment els protocols diferents disponibles són: TFTP, HTTP, FTP i csend/crecieve. De totes maneres, hi ha malware que utilitza protocols propis per la propagació. També hi ha alguns mòduls per aquests casos. Aquests mòduls poden ser desactivats, ja que contactar una màquina d'internet per a descarregar informació pot ser no ètic i més pensant que la majoria seran ordinadors infectats.

Submission modules

Finalment els *submission modules* controlen els fitxers descarregats. Ara mateix hi ha quatre diferents tipus de mòduls:

- Un mòdul que guarda el fitxer en una localització configurable al sistema de fitxers permetent el canvi de propietari.

- Un mòdul que enregistra el fitxer en una base de dades central.
- Un mòdul que enregistra el fitxer a un altra instància nephes per activar la estructura d'herència dels sensors nepenthes.
- Un mòdul que enregistra el fitxer en un sistema antivirus per ser analitzat posteriorment.

3.2.2 Instal·lació i configuració

Per a la instal·lació de la plataforma nepenthes, hi ha paquets autoinstal·lables per a les distribucions:

- Gentoo
- Debian
- OpenBSD
- FreeBSD

També estan disponibles les fonts que permeten l'ús de l'automake per a comprovar si el sistema satisfà els requisits de llibreries:

- g++
- libcurl
- libmagic
- libpcre
- libadns
- flex i bison

Un cop instal·lat, és recomanable editar el fitxer de configuració nepenthes.conf i revisar totes les configuracions dels mòduls per tal d'assegurar el bon funcionament de nepenthes.

Un cop executat podem comprovar que realment estem oferint vulnerabilitats:

```
 [root@senbei ~]# lsof -i
COMMAND    PID    USER   FD   TYPE DEVICE SIZE NODE NAME
portmap   1435     rpc    3u  IPv4   4210       UDP *:sunrpc
portmap   1435     rpc    4u  IPv4   4211       TCP *:sunrpc (LISTEN)
rpc.statd 1454 rpcuser    3u  IPv4   4240       UDP *:filenet-tms
rpc.statd 1454 rpcuser    6u  IPv4   4227       UDP *:782
rpc.statd 1454 rpcuser    7u  IPv4   4251       TCP *:34574 (LISTEN)
hpiod     1673    root    0u  IPv4   4646       TCP senbei:50000 (LISTEN)
python    1678    root    4u  IPv4   4678       TCP senbei:50002 (LISTEN)
cupsd     1689    root    0u  IPv4   4700       TCP senbei:ipp (LISTEN)
cupsd     1689    root    2u  IPv4   4701       UDP *:ipp
sshd      1698    root    3u  IPv6   4729       TCP *:ssh (LISTEN)
sendmail  1716    root    4u  IPv4   4822       TCP senbei:smtp (LISTEN)
avahi-dae 1855   avahi   13u  IPv4   5159       UDP *:mdns
avahi-dae 1855   avahi   14u  IPv4   5160       UDP *:filenet-rpc
avahi-dae 1855   avahi   15u  IPv6   5161       UDP *:filenet-nch
dhclient  2643    root    5u  IPv4   7439       UDP *:bootpc
nepenthes 2648    root    3u  IPv4   7515       UDP *:filenet-rmi
nepenthes 2648    root    6u  IPv4   7516       TCP :smtp (LISTEN)
nepenthes 2648    root    7u  IPv4   7517       TCP :pop3 (LISTEN)
nepenthes 2648    root    8u  IPv4   7518       TCP :imap (LISTEN)
nepenthes 2648    root    9u  IPv4   7519       TCP :imap3 (LISTEN)
nepenthes 2648    root   10u  IPv4   7520       TCP :smtps (LISTEN)
nepenthes 2648    root   11u  IPv4   7521       TCP :imaps (LISTEN)
nepenthes 2648    root   12u  IPv4   7522       TCP :pop3s (LISTEN)
nepenthes 2648    root   13u  IPv4   7523       TCP :urbisnet (LISTEN)
nepenthes 2648    root   14u  IPv4   7524       TCP :6129 (LISTEN)
....
```

Tot seguit podem configurar el nostre honeypot per a què ens reporti informació mitjançant un bot del IRC:

```
 submit-norman
{
        // this is the adress where norman sandbox reports will be sent
        email   "my.email@example.com";
};
```

També es pot editar la configuració per a què un cop capturada informació, nepenthes enviï la captura al Norman's sandbox, qui efectuarà una anàlisi i enviarà una còpia al nostre email.

Aixi doncs, un cop el malware ataqui el sensor, Nepenthes intentarà descarregar el codi i l'enviarà al Norman sandbox. A continuació es detalla una mostra de la informació aportada per un bot de l'IRC:

```
 [ Network services ]
   * Looks for an Internet connection.
   * Connects to xxx.example.net on port 7654 (TCP).
   * Sends data stream (24 bytes) to remote address xxx.example.net,
 port 7654.
   * Connects to IRC Server.
   * IRC: Uses nickname xxx.
   * IRC: Uses username xxx.
   * IRC: Joins channel #xxx with password xxx.
   * IRC: Sets the usermode for user xxx to ...
```

Alguns tipus de malware poseeixen sistemes anti-debug i que eviten que Norman ens aporti informació útil. Llavors és quan es recomana enviar la captura a un antivirus, com pot ser VirusTotal, que ens escanejarà el que li reportem amb 20 tipus diferent de software.

Els binaris capturats, són guardats a /opt/nepenthes/var/binaries:

```
# ls /opt/nepenthes/var/binaries
01a7b93e750ac9bb04c24c739b09c0b0  547765f9f26e62f5dfd785038bb4ec0b
99b5a3628fa33b8b4011785d0385766b  055690bcb9135a2086290130ae8627dc
54b27c050763667c2b476a1312bb49ea  ...
```

També hi ha logs de cada descàrrega feta:

```
 tail -1 /opt/nepenthes/var/log/logged_submissions
[2006-07-05T20:37:52]
ftp://ftp:password@xxx.info:21/host.exe
eb6f41b9b17158fa1b765aa9cb3f36a0
```

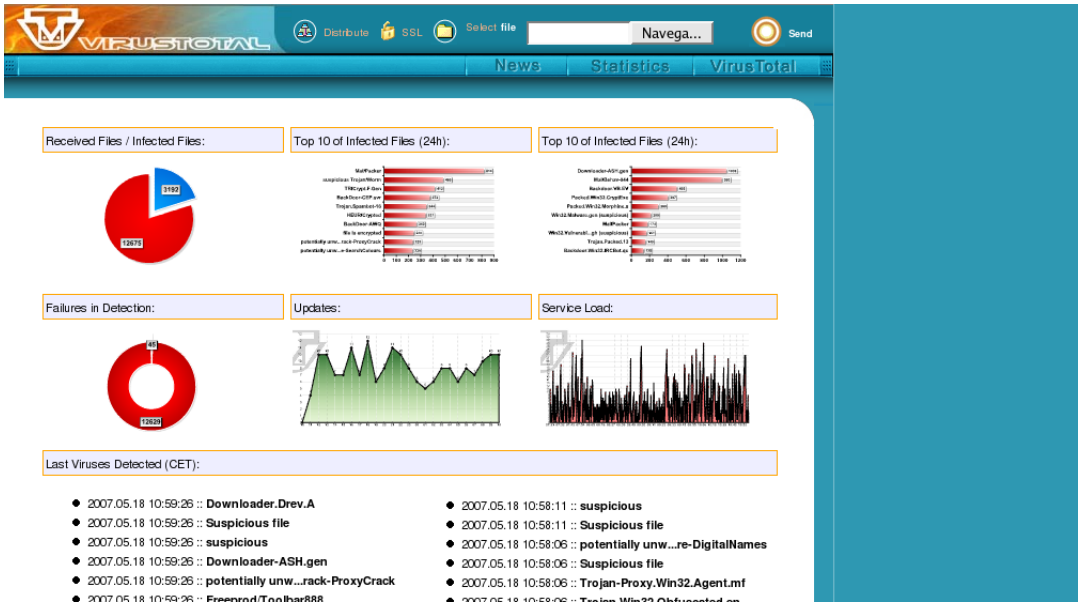


Figura 3.12: Pàgina de VirusTotal

3.2.3 Anàlisi

Nepenthes ens aporta diversos sistemes de d'avís a l'hora de trobar nous atacs que ens facilitaran la tasca de l'anàlisi de la seguretat en xarxes mitjanes i grans.

Tot i això s'han trobat diverses limitacions:

1. Només es poden capturar atacs que es propaguen automaticament
2. És difícil capturar atacs que utilitzen una llisat d'objectius per trobar sistemes als quals atacar
3. És possible detectar l'ús de Nepenthes, ja que aquest normalment emula un gran nombre de ports TCP que l'atacant pot detectar rapidament durant la fase de reconeixement.

Avantatges de Nepenthes	Desavantatges de Nepenthes
Gran quantitat de vulnerabilitats	Díficil configuració i manca de documentació
Es gratuït i de codi lliure, serà desenvolupat ràpidament amb l'ajuda de la comunitat	No proporciona support formal
Tot i ser d'interacció baixa pot aportar molta informació	La simulació de les mateixes vulnerabilitats el delata

Taula 3.2: Avantatges i desavantatges de Nepenthes

3.3 Sebek

Des que la filosofía honeypot va ser posada a la pràctica s'ha intentat recopilar tota la informació possible de les accions de l'intrús amb la màxima discreció possible. A més a més, apareix un obstacle més: dades xifrades. Si un atacant es connecta a la nostra màquina utilitzant SSH, o qualsevol altra eina que faci crides al sistema per mitjà de dades xifrades la feina d'anàlisi d'un honeypot enverinat es complica molt.

Així doncs, Sebek es presenta com la solució idònia als nostres problemes. Val a dir que Sebek no es pot definir com un honeypot *per se*, sinò que amb la combinació d'una màquina amb o sense debilitats afegides pot esdevenir una font d'informació molt bona.

Sebek és una eina de recollida de dades que funciona dins del nucli del sistema per a recollir la informació dels events que succeixen. Quan les dades no estan xifrades, podem saber què passa dins del nostre honeypot recollint tràfic d'entrada i de sortida i revisant tots els fitxers utilitzats durant la sessió, ara bé, quan les dades estan xifrades aquests recursos esdevenen inútils ja que l'obtenció de la clau per desxifrar pot arribar a ser impossible. La idea en la qual es basa Sebek és obtenir les dades en el post-desxiframent un cop les ordres hagin de ser transmeses al kernel per què aquest les pugui entendre. Es llavors quan la informació tindrà més valor ja que podrem entendre-la.

Després Sebek enviarà aquestes dades a un servidor de manera oculta.

En una primera versió, Sebek només era capaç d'obtenir les comandes enviades al kernel. En la segona versió totes les dades són capturades, des d'un fitxer de text fins tota una conversa d'IRC de l'intrús.

Aquesta eina va ser desenvolupada des d'un principi per la comunitat de Honeynet Project, actualment està en la versió 2 tot i que ens referirem a ella igualment com Sebek.

3.3.1 Arquitectura

Sebek consta de dos components: un servidor i un client. El client obté les dades del honeypot mentre que el servidor captura les dades enviades a la xarxa per el client. El servidor pot actuar de dues maneres:

1. agafar les dades al moment que les envia el client, és a dir, en viu.

2. captura de les dades en format tcpdump un cop acabada la sessió.

Les connexions són enviades per UDP sense cap protocol de capa superior que proporcioni control de la comunicació.

Client Sebek

El client resideix en el nucli, i en el cas de Linux ha estat implementat com un *Loadable kernel Module* (LKM). Pot recollir tota la informació mitjançant la crida al sistema `read()`. Sebek fa això substituïnt la funció `read()` a la `System Call Table` per una nova. La nova funció només crida a la vella i copia el contingut a un buffer.

Quan un procés fa la crida a sistema `read()` el que succeirà és que s'executarà la crida que ha modificat Sebek, en aquest punt tindrà accés a totes les dades que hagin estat enviades al kernel mitjançant aquesta crida.

Sebek també s'autoamaga, de manera que no pugui ser vist per l'usuari del sistema. Per aquest motiu s'instal·la un segon mòdul anomenat *the cleaner*. Aquest mòdul manipula la llista dels mòduls que hi ha al sistema de manera

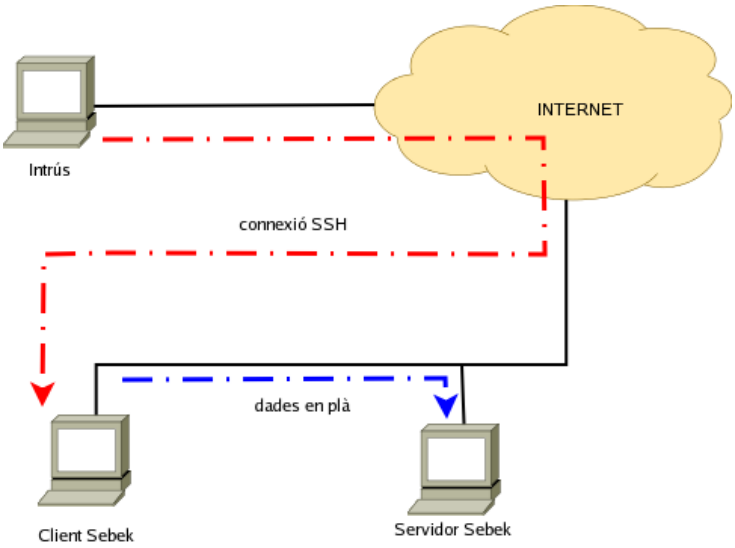


Figura 3.13: Funcionament de l'arquitectura client-servidor de Sebek

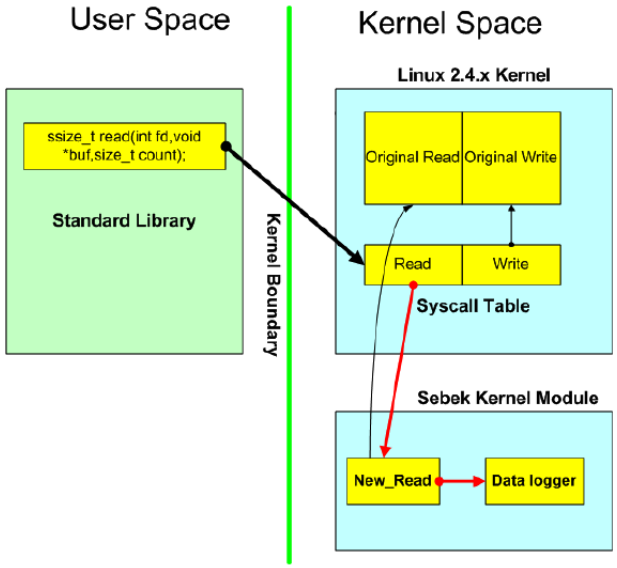


Figura 3.14: Substitució de la crida al sistema

que el Sebek és eliminat. Per aquest motiu apareixen dos problemes: un és que l'usuari no sabrà si Sebek està instal·lat en el sistema ja que no hi haurà

rastre d'ell, i la segona és que no es podrà eliminar el mòdul (`rmmod`). Per aquest mateix motiu, Sebek permet no amagar el mòdul durant la fase en la que estem fent proves per tal de poder treure'l.

Un cop hem capturat les dades de l'intrús, hem d'enviar-les al servidor de manera que l'atacant no s'adoni que les dades estan essent transmeses. Tot i que la xarxa LAN no és la manera més segura de transmetre les dades, serà utilitzada per aquesta finalitat, però si el client simplement enviés els paquets UDP al servidor l'intrús podria detectar aquest tràfic i comprovar que realment hi ha un Sebek funcionant. A pesar d'aquest inconvenient Sebek enviarà dades en un flux UDP, tanmateix, abans de fer tal cosa, modifica el nucli del sistema per a què no pugui detectar els paquets Sebek generats pel client ni cap altre paquet Sebek de la xarxa. Després, quan Sebek transmet dades s'assegura que el sistema no pugui blocar la transmissió ni tan sols contar els paquets. Si tots els honeypot en la xarxa LAN tenen Sebek instal·lat, cap d'ells pot veure les dades transmeses cap al servidor. Podem assegurar que hem creat un canal segur per a la transmissió de dades aconseguides pels clients.

Servidor

Com ja hem dit abans, el servidor pot recollir les dades de dues maneres: la primera és recollir les dades d'un fitxer de log en format tcpdump. I la segona és capturar el tràfic en viu mentre l'intrús és dins del honeypot.

El servidor consta dels següents components: `Sbk_extract` que captura les dades en format tcpdump o directament del client (com un sniffer). `sbk_extract` obté les dades, aquestes poden ser tractades de dues maneres:

1. utilitzar l'eina `sbk_ks_log.pl` que recull les commandes de l'intrus i l'envia per la sortida estandar.

2. `sbk_upload.pl` emmagatzema tota la informació proporcionada pels clients en una base de dades MySQL.

3.3.2 Anàlisi

Sebek és una eina de captura de dades ubicada en el nucli del sistema. Està dissenyada per a capturar tota l'activitat en el honeypot de manera encoberta. El problema del xifrat de dades es veu solucionat capturant l'activitat que ocorre en el kernel, on es enviada desxifrada. Per aquest motiu es poden capturar comandes executades, passwords i monitoritzar qualsevol comunicació, ja sigui chats d'IRC, email i activitat SSH/SFTP. En general Sebek proporciona un punt de vista exepcional de tot el que ocorre dins d'un honeypot.

La versió actual de Sebek té certs límits, ja que per a un intrús amb suficient nivell del Sistema Operatiu que corre dins de la màquina hi ha maneres de detectar la presència de Sebek. Tanmateix, no està clar si es pot fer alguna cosa al respecte. Això es podria evitar sent lo suficientment subtils per no aixecar sospites i no fer que l'atacant pugui pensar que res està succeïnt dins aquella màquina.

El treball futur de Sebek no és intentar amagar encara més el funcionament d'aquest sinò la capacitat de recollir encara més dades per a millorar l'anàlisi posterior.

3.4 *Homemade* honeypot

Fins ara hem vist tres possibles sol·lucions per a l'implementació d'un honeypot, les tres són paquets instal·lables que facilment podem construir en el nostre ordinador. El que ara farem serà discutir com nosaltres podem construir el nostre propi honeypot, amb la qual cosa aconseguirem una eina que s'adapta molt més al que nosaltres volem aconseguir. A més, aconseguirem controlar molt més tot el que succeeix en cada atac ja que tindrem control total sobre les tecnologies involucrades.

Construir el teu propi honeypot casolà no és tant complicat com pot semblar a primera vista. Utilitzant algunes eines, una mica de codi i molta imaginació es poden aconseguir sol·lucions a mida per cada cas. Per exemple, podem voler esbrinar una sèrie de proves o escannejos o per contra, es pot voler deixar una màquina completament oberta simulant un sistema en producció per recopilar molta informació. Tot seguit es presenten tres possibles sol·lucions:

3.4.1 Monitorització de ports

Aquest és el tipus més sencill de *homemade* honeypot, simplement reportarà certa informació que escolta d'un port determinat. Com més ports escoltem, més informació obtindrem del que s'intenta connectar al nostre honeypot.

La monitorització de ports és molt flexible en termes d'informació recol·lectada. La seva utilització bàsica és la de detecció. Podem detectar proves o altres tipus d'activitat no autoritzada. Com més serveis estiguem escoltant, més patrons d'atac podrem detectar. A més és excel·lent per complementar altres honeypots en l'obtenció d'informació de serveis que no es puguin emular.

Un dels serveis més atacats són els serveis web, més específicament el port 80 (HTTP). Aquest port és utilitzat per moltes organitzacions. Els servidors web difícilment poden detectar un atac degut a la gran quantitat d'informació que gestionen. A més els NIDS no són capaços encara de detectar un patró

d'atac HTTP. Això comporta que l'administrador del servei web no s'adoni de què algun cuc l'ha infectat fins que un client es queixa d'un servei lent.

Els honeypots per una altra banda, ràpidament poden informar d'un patró d'atac. Com que no tenen cap funció productiva, el tràfic que sigui enviat al seu port 80 serà etiquetat de sospitós. En pocs minuts es podrà saber si algun tipus de malware està intentant infectar els nostres servidors. També ens pot informar de certs patrons d'atac per una possible infecció.

Hi ha moltes maneres de fer una escolta de ports, com hem vist si volem escoltar un port i capturar el virus que ens intenta infectar o dades de l'intrús, haurem de simular el servei corresponent. Per altra banda, si només volem capturar els intents de connexió, amb una *port-monitornig tool* ens serà suficient.

`Netcat` és una eina molt versàtil desenvolupada per plataformes Unix. Netcat crea connexions TCP i UDP entre sistemes permetent el traspàs d'informació mitjançant `pipes`. Per exemple, si volem veure que passa en el port 80 del nostre honeypot casolà per detectar qualsevol connexió:

```
[root@idefix ~]# nc -l 80
GET / HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; U; Linux i686; ca; rv:1.8.0.10)
Gecko/20070223 Fedora/1.5.0.10-1.fc5 Firefox/1.5.0.10
Accept: text/xml,application/xml,application/xhtml+xml,
text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: ca,en-us;q=0.7,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
```

Es pot veure com el netcat ha capturat un intent de connexió amb un explorador d'internet.

EL risc associat amb aquest tipus de honeypot és baix, ja que es tracta d'un *low-interaction honeypot*. De totes maneres hem de comptar amb un sistema segur ja que pot ser que es vegui sotmés a molts tipus d'atacs.

3.4.2 Chroot (entorns de gàbia)

Ara ens encarregarem d'estudiar un tipus més complex de honeypots casolans. Són les anomenades gàbies o entorns chroot, també coneguts com *change root*. Es tracta d'un honeypot d'interactivitat mitja capaç d'emmagatzemar grans quantitats d'informació. Els entorns de gàbia simulen un sistema operatiu amb certes limitacions que faran que l'intrús cregui que està dins d'un sistema sencer, evitant en tot moment que pugui fer-se amb el control de la màquina.

Tot i que els entorns de gàbia van ser desenvolupats com un mecanisme de seguretat, no va ser per a l'ús de honeypots. Els chroot vàren estar pensats per garantir la seguretat de serveis vulnerables en les organitzacions. Com a exemple, tenim els serveis de DNS i Web, que són serveis accessibles per tothom des de qualsevol punt d'Internet i que comporten un cert risc quan a seguretat es refereix. És preferible mantenir aquest servei en un entorn aïllat sense exposar un sistema sencer al perill de ser vulnerat.

Com a honeypots, les gàbies tenen dos principals avantages: la flexibilitat i la capacitat de poder simular qualsevol servei funcionant dins d'un sistema. Podem aconseguir fer funcionar qualsevol servei que hi hagi en el nostre sistema en producció. Un altre avantatge és la capacitat d'interacció amb el sistema engabiat.

Gràcies a la seva flexibilitat, aquests tipus de honeypot es poden utilitzar en la recerca i en la producció. En la producció, els podem utilitzar per la prevenció i per l'engany. És possible que un atacant estigui dins del nostre sistema engabiat pensant que ha vulnerat un cert servei, però del que no se'n adona és que està essent monitoritzat i que totes les accions que dugui a terme seran reportades a l'administrador. A la vegada, l'estarem distraient i fent que perdi el temps i els recursos en un lloc on no ens pot causar problemes.

Respecte a la recerca, cada atac que comprometi el sistema ens aportarà la suficient informació per a poder prevenir i actuar en atacs posteriors.

Els entorns de gàbia utilitzen la comanda `chroot` per crear l'entorn. Conceptualment, `chroot` funciona creant un nou directori arrel per al procés. Aquest procés només pot accedir als fitxer dins del nou directori arrel. Qualsevol nou atac, només podrà accedir als fitxers que nosaltres vulguem.

3.4.3 Màquines virtuals

La virtualització es refereix l'abstracció de recursos en un computador. Podem virtualitzar uns certs recursos o una plataforma sencera. En aquesta secció explicarem com podem aprofitar els avantatges que ens ofereix la virtualització per a construir un honeypot prou segur i eficient.

L'eina de virtualització més coneguda arreu és el VMWare. Es tracta d'una *siute* de software que ens permet virtualitzar un cert nombre de sistemes operatius basats en arquitectura Intel. Amb aquest software serem capaços d'executar sistemes operatius com Windows, Linux, BSD, Solaris, etc. en una màquina. Existeixen també eines lliures per a la virtualització. Una de les més utilitzades és Xen, que a diferència de VMWare i una altra eina com és VirtualPC (que solen utilitzar molts recursos i baixar el rendiment de la màquina) utilitza nuclis modificats. Això fa que es puguin millorar certes característiques, cosa que permet tenir tots els beneficis de la virtualització sense haver de sofrir gaire sobreutilització de recursos. Per tant es poden fer funcionar sistemes operatius a una velocitat pròxima a l'habitual.

A l'hora de fer la virtualització, una de les coses a tenir en compte serà el funcionament del disc dur:

- **Disc virtual:** Es tracta d'un fitxer que farà de disc.
- **Partició RAW:** Ofereixen una partició real en un disc real.

Depenent de la utilització que li donguem a la màquina virtual haurem d'utilitzar un o altre disc. Si el que volem és un honeypot per a la prevenció,

la virtualització ens permetrà una gran interacció si som atacats, a la vegada que se'ns simplifica la instal·lació. Ara bé, si el sistema operatiu es veu força danyat, la tasca forense a l'hora de seguir els pasos no serà gens fàcil, ja que no podrem utilitzar eines automàtiqües i l'accés al disc serà difícil.

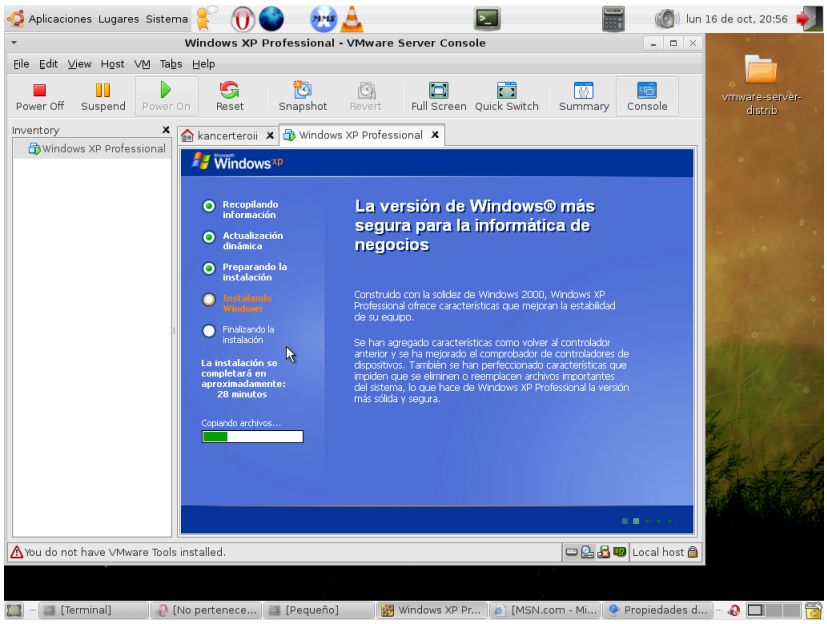


Figura 3.15: Execució de WinXP en un Ubuntu utilitzant VMWare

Si la nostra previsió és que el honeypot sofreixi grans atacs, la utilització de particions ens donarà més llibertat a l'hora d'accedir a les dades, fins i tot es recomana tenir un disc físic separat per a utilitzar com disc de la màquina virtual.

Un cop estiguem utilitzant la nostra màquina virtual, l'haurem de preparar per a poder visualitzar la informació recollida. Com ja sabem, algun tipus de malware (com el CodeRed) resideix només en memòria. La utilització de màquines virtuals per a l'ús de honeypots facilita la tasca a l'hora de fer un examen forense.

Una de les tècniques utilitzades és obligar l'atacant a fer servir la memória d'intercanvi: fent que el disc de la màquina virtual forci l'atacant a utilitzar la memòria *swap*. Això ens donarà l'oportunitat de tenir-lo emmagatzemat per

a una anàlisi posterior. Una altra característica que aporten les màquines

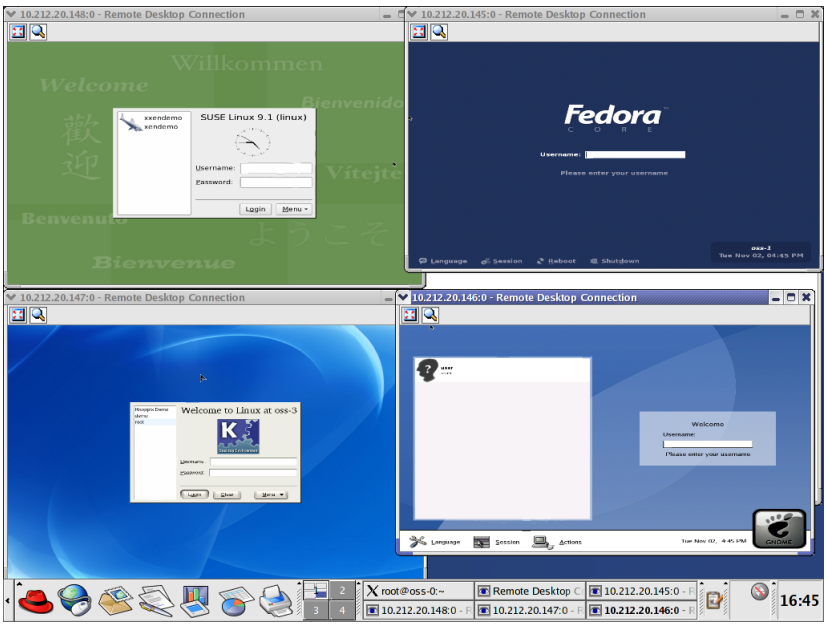


Figura 3.16: Execució de 5 Linux diferents utilitzant Xen

virtuals és la capacitat de suspendre el sistema operatiu. És similar a la hivernació que fan els ordinadors portàtils quan se'ls esgota la bateria. Quan se suspén la màquina, una imatge de la memòria RAM es guardada en el disc. Un cop emmagatzemat, aquest fitxer pot ser accedit per a mirar la informació continguda. Un cop analitzada la informació podem seguir executant la màquina virtual.

3.4.4 Anàlisi

Com s'ha pogut comprovar, la contrucció d'un honeypot no requereix grans coneixements de programació, l'ingeni es qui predomina en aquest camp. Com ja s'ha pogut veure en seccions anteriors l'utilització de diferents eines en un mateix honeypot augmentara la sotisficació d'aquest. Així doncs podem construïr el nostre honeypot casolà combinant diferents eines i diferents tècniques per a obtenir la informació necessària. Un cop acabada la nostra

feïna aconseguirem un honeypot adaptat a les nostres necessitats que podrem controlar al nostre parer.

El risc associat a aquest tipus de honeypots resideix, com de costum, en la interactivitat oferida. A banda d'aquesta trivialitat, també haurem de ser capaços de controlar les eines utilitzades, ja que una mal configuració pot ser devastadora per als nostres interesos.

Avantatges	**Desavantatges**
Escalabilitat i adaptació a les nostres necessitats	Requereixen molta dedicació i estudi
Utilització del programari que nosaltres desitjem	Dificultat per recuperar la informació de l'atac
Gran ventall d'interactivitat (a la nostra elecció)	Risc associat elevat

Taula 3.3: Avantatges i desavantatges dels *Homemade* Honeypots

Capítol 4

Avaluació de la xarxa

En aquest capítol tractarem d'observar quines estructures de xarxa ens trobarem a l'hora d'implementar les sol·lucions de seguretat amb els honeypot. Ho farem fent una observació superficial de l'arquitectura avaluant uns quants detalls per poder adaptar cada honeypot a les nostres necessitats.

En primer lloc, analitzarem els elements que hem de tenir en comte:

4.1 Estructura de la xarxa

La xarxa de la UdL dóna servei a sis campus dispersos per la ciutat de Lleida:

- Rectorat
- Agrònoms
- Cap-pont
- Ciències de la salut
- Caparrella
- Unitat Docent Hospital Arnau de Vilanova

La interconnexió dels campus es fa utilitzant una topologia d'estrella amb centre a Rectorat i emprant enllaços Gigabit Ethernet de fibra òptica monomode. En cada campus s'utilitza també una topologia d'estrella per donar

servei als diferents edificis. Dins de cada campus s'empra igualment tecnologia Gigabit Ethernet però en aquest cas sobre enllaços de fibra òptica multimode. L'equipament de xarxa està composat principalment per commutadors de nivell d'enllaç. La connexió amb l'exterior (Internet) s'efectua mitjançant un enllaç de 200 Mbps situat en Rectorat. En la figura 4.1 es mostra l'estructura física de la xarxa de la UdL.

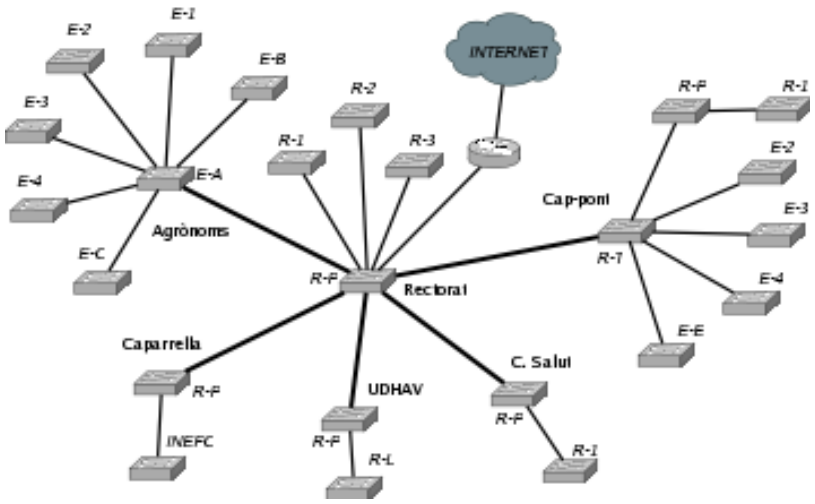


Figura 4.1: Estructura de la xarxa de la UdL

Actualment la xarxa té 6500 punts de cablatge estructurat FTP de categoria 5 dels cuals 1500 estan destinats a telefonia i 3400 a punts de dades. Per tal de donar servei als diferents tipus d'usuaris i separar el màxim posible els diferents tipus de trànsit, s'han creat 19 xarxes virtuals (VLAN) que s'estenen per els diferents edificis en funció de les necesitats. D'aquestes 19 VLAN sol s'han considerat addients per efectuar l'estudi quatre:

- Intranet
- Aules
- Wifi
- DMZ

Aquestes VLAN són les més importants de la xarxa, a continuació es descriuen les característiques més rellevants de cadascuna.

4.1.1 Intranet

Es tracta de la VLAN amb més usuaris de la xarxa ja que s'hi connecten la majoria de servidors interns i dóna servei al PAS i al Professorat. L'adreçament assignat a aquesta VLAN és la classe A 10.0.0.0 i en l'actualitat té 2200 punts actius. Per a la sortida a internet es fa servir *masquerading* a través dels firewalls, tot i que alguns serveis surten amb adreça IP publica mitjançant NAT (*Network Adress Translation*) en el firewall. Per poder analitzar millor aquesta VLAN es descriuran les carateristiques dels seus usuaris:

Servidors interns

Equips administrats i supervisats per l'ASIC (Àrea de Sistemes i Comunicacions), les dades que contenen són molt importants i estan fortament protegits contra atacs de qualsevol característica. El sistema operatiu predominat en els servidors és Linux, la qual cosa comporta un grau més de seguretat. Alguns serveis són accesibles des de l'exterior, és aquí on es concentren la majoria de serveis que tenen IP pública emprant NAT.

PAS

Degut al pla de migració a software lliure un 40% dels equips tenen instal·lat el sistema operatiu Linux mantingut per ASIC, això dóna un plus de seguretat ja que les aplicacions que s'executen en cada màquina estan controlades, així com l'ús que se'n pot fer d'elles. La resta d'equips tenen instal·lat Windows XP, la qual cosa fa que el tràfic de bradcast augmenti gràcies als paquets NETBios. En no poder controlar les aplicacions que s'executen en cada màquina amb Windows el risc augmenta considerablement.

Professorat

Els equips dels profesorat són molt heterogenis tant pel que fa al hardware com al software. Els sistemes operatius predominants són Windows, Linux i MAC. Cal comtemplar que també connecten a la xarxa portàtils que també sovint són conectats a diverses xarxes. Són una font de problemes, principalment virus i malware.

4.1.2 Aules

Aquesta VLAN interconnecta totes les aules de docència de la UdL i actualment té 1000 punts actius. Per aquesta VLAN s'empra la classe B 172.16.0.0/16. La interconnexió amb la resta de la UdL està supervisada per un firewall. El sistema operatiu predominant és Windows encara que hi ha sobre un 20% de Linux i un percentatge molt baix de Apple Macintosh (MAC). Tot i això el programari utilitzat està força controlat, i la manera habitual de connexió es l'autenticació dels usuaris que facilita el control de l'ús dels equips. Els usuaris són alumnes i cal tenir en compte que hi ha una escola d'Informàtica, sovint poden ser un focus de problemes.

4.1.3 Wifi

VLAN emprada per interconnectar tots els punts d'accés que donen cobertura Wifi. Estan associats a Eduroam i els usuaris poden navegar sol si són autenticats mitjançant un portal captiu. Hi ha un total de 65 punts d'accés repartits per tots els campus, cada punt d'accés pot arribar a donar connexió a uns 35 o 40 usuaris, el que comporta un potencial aproximat de 2000 usuaris.

4.1.4 DMZ

VLAN desmilitaritzada per la connexió a Internet i on se situen alguns servidors totalment accesibles des de l'exterior. Està situada entre el router de

sortida a Internet i el firewall principal de la xarxa. Les IP són públiques i el tràfic és molt divers.

En la figura 4.2 es mostra l'estructura lògica de les VLAN descrites anteriorment

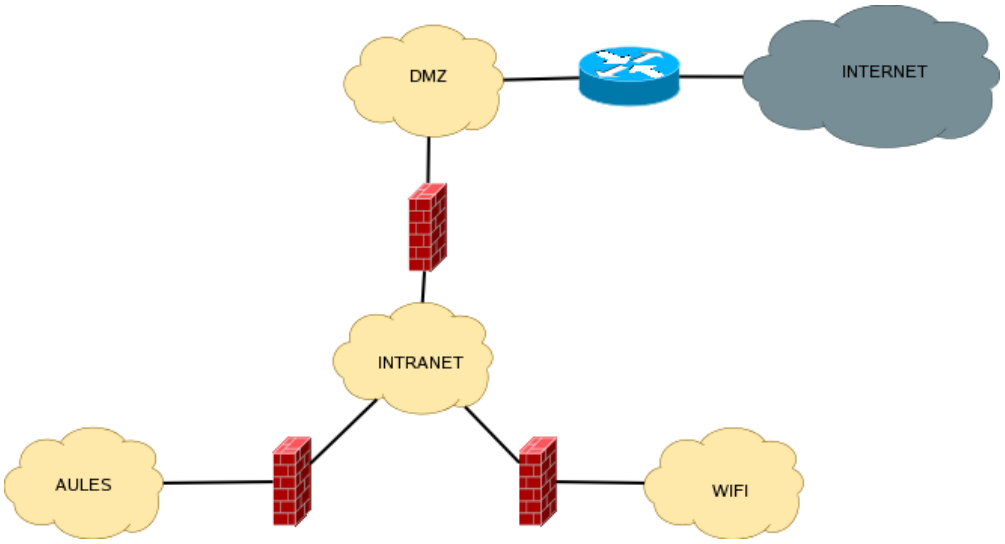


Figura 4.2: Esquema logic de les VLAN de la xarxa de la UdL

L'esquema de la figura 4.2 ens mostra una simplificació de la xarxa de la Universitat de Lleida.

Cal tenir en compte que podem diferenciar entre dos tipus de xarxes:

- **internes**: protegides darrera del firewall, en les que els usuaris són coneguts.
- **externes**: fora del firewall, on qualsevol persona hi pot accedir.

Cada característica ens portarà a ubicar en cada xarxa una implementació adaptada a la informació que vulguem recollir. Així doncs, tenint en compte la ubicació de cada VLAN en la xarxa de la UdL i les particularitats dels usuaris, maquinari i programari s'haurà d'adaptar una solució òptima que ens aporti la màxima infarmació possible amb el màxim valor.

Capítol 5

Implementació

Abans de començar a instal·lar o desenvolupar qualsevol honeypot, hem de determinar que és el que volem arribar a aconseguir. Un cop haguem decidit els nostres objectius, seleccionarem el honeypot més adecuat per implementar en la nostra organització.

5.1 Objectius

Després d'haver fet l'avaluació de la xarxa estem preparats per a acotar els nostres objetius en cadascuna de les VLAN de la xarxa de la UdL:

Intranet

Com es tracta d'una vlan molt segura, en la que difícilment tindrem atacs de l'exterior, ens interessarà detectar atacs automatitzats de virus i malware. Com que es tracta d'una xarxa que conté molta informació, els logs de proves a alguns ports no ens ajudaran a determinar cap atac. Així doncs, el més apropiat serà instal·lar un honeypot Nepenthes, el qual ens aportarà informació de virus al moment (per mitjà de correu eletrònic o bots d'IRC) i a més ens capturarà una mostra del virus per tal de poder subsanar els danys causats.

Alumnes

Pel que sabem, és una vlan molt controlada ja que els equips que hi estàn connectats estan tots localitzats i mantinguts per l'administrador, així doncs continuem tenint el perill de què els virus intensifiquin el trànsit de la xarxa i a més pot ser que rebem alguna mena d'atac per part dels usuaris. Al igual que la intranet, l'eina més apropiada contínua essent Nepenthes tot i que també es podria afegir més interactivitat per la captura d'atacs no automatitzats. Malgrat això no són freqüents.

Wifi

Desconeixem el que trobarem, ja que els equips que es conectaran a aquesta xarxa són equips d'usuari no mantinguts per l'administrador, per tant el més apropiat podrà ser un honeypot que ens monitoritzés certes proves per analitzar patrons d'atac. L'eina més adient seria HoneyD, ja que per molt que puguem capturar els virus amb la plataforma Nepethes no podrem actuar al respecte.

DMZ

Serà la xarxa on tindrem més possibilitats d'atacs de tot tipus, haurem d'utilitzar tot el nostre enginy per a desenvolupar un o més honeypots per detectar tot típus d'atac.

5.2 Equipament

Tot el desenvolupament del treball, tant d'anàlisi com d'implementació, s'ha dut a terme en dues màquines cedides per l'àrea CCIA (Ciències de la Computació i Intel·ligència Artificial) de la EPS. Tot i que són dos ordinadors força ben equipats, no és necessari l'ús de màquines molt potents per a l'implementació dels honeypots. Tot seguit es mostren les caraterístiques d'aquests ordinadors.

Arale
```
[root@arale ~]# cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 15
model           : 2
model name      : Intel(R) Pentium(R) 4 CPU 2.00GHz
stepping        : 4
cpu MHz         : 2000.062
cache size      : 512 KB

[root@arale ~]# free
total       used       free     shared    buffers     cached
Mem:   515096     505700       9396        0       97600     236720
-/+ buffers/cache:     171380     343716
Swap:      1048568        120    1048448
```

Senbei
```
[root@senbei ~]# cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 15
model           : 2
model name      : Intel(R) Pentium(R) 4 CPU 2.00GHz
stepping        : 4
cpu MHz         : 1999.895
cache size      : 512 KB

[root@senbei ~]# free
total       used       free     shared    buffers     cached
Mem:   1034992     635244     399748       0       64876      459044
-/+ buffers/cache:     111324     923668
Swap:      2031608          0    2031608
```

S'han utilitzat també com a equipament de xarxa:

OmniSwitch Xylan: Durant la fase de proves va servir per a crear diverses VLAN i configurar el Honeyd en un entorn aïllat.

Alcatel OmniAcces 512: Va donar-nos accés a les VLAN en producció de la UdL durant la fase d'analisi.

Figura 5.1: Equipament de xarxa a l'Aula Alcatel (1)

Alcatel OmniSwitch 6602: Utilitzat durant la fase d'implementació de tot el projecte.

5.3 Implementació i configuració de Nepenthes

La configuració de Nepenthes ha estat la mateixa en les tres VLAN on l'hem posat en marxa:

- Intranet
- Alumnes
- Wifi

A continuació es mostren les vulnerabilitats configurades:

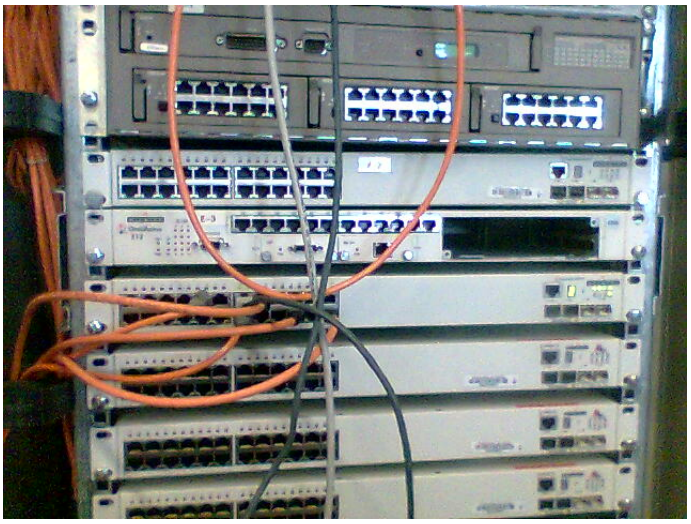


Figura 5.2: Equipament de xarxa de l'Aula Alcatel (2)

```
// moduls de vulnerabilitat
   "vulnbagle.so",                "vuln-bagle.conf",              ""
   "vulndameware.so",             "vuln-dameware.conf",           ""
   "vulndcom.so",                 "vuln-dcom.conf",               ""
   "vulnftpd.so",                 "vuln-ftpd.conf",               ""
   "vulniis.so",                  "vuln-iis.conf",                ""
   "vulnkuang2.so",               "vuln-kuang2.conf",             ""
   "vulnlsass.so",                "vuln-lsass.conf",              ""
   "vulnmsmq.so",                 "vuln-msmq.conf",               ""
   "vulnmsdtc.so",                "vuln-msdtc.conf",              ""
   "vulnmssql.so",                "vuln-mssql.conf",              ""
   "vulnmydoom.so",               "vuln-mydoom.conf",             ""
   "vulnnetbiosname.so",          "vuln-netbiosname.conf",        ""
   "vulnnetdde.so",               "vuln-netdde.conf",             ""
   "vulnoptix.so",                "vuln-optix.conf",              ""
   "vulnpnp.so",                  "vuln-pnp.conf",                ""
   "vulnsasserftpd.so",           "vuln-sasserftpd.conf",         ""
   "vulnsub7.so",                 "vuln-sub7.conf",               ""
   "vulnupnp.so",                 "vuln-upnp.conf",               ""
   "vulnveritas.so",              "vuln-veritas.conf",            ""
   "vulnwins.so",                 "vuln-wins.conf",               ""
```

```
    "vulnasn1.so",                  "vuln-asn1.conf",               ""
```

Per a què Nepenthes reporti la informació rebuda es va executar en mode debug i es va configurar el mòdul del bot de IRC:

```
 log-irc
{
    use-tor     "0";
    tor
    {
        server  "localhost";
        port    "9050";
    };

    irc
    {
        server
        {
            name    "irc.fef.net";
            port    "6667";
            pass    "";
        };

        user
        {
            nick        "senbei";
            ident       "nepenthes";
            userinfo    "http://nepenthes.mwcollect.org/";
            usermodes   "+i";
        };

        channel
        {
```

```
            name    "#vila-pingui";
            pass    "foo";
        };
```

Un cop nepenthes està en funcionament aquest es l'aspecte que té:

```
 Starting Nmap 4.03 ( http://www.insecure.org/nmap/ )
at 2007-06-06 17:16 CEST
Interesting ports on aules-102-080.udl.net (172.16.102.80):
(The 1651 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
25/tcp    open  smtp
42/tcp    open  nameserver
80/tcp    open  http
110/tcp   open  pop3
111/tcp   open  rpcbind
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
143/tcp   open  imap
220/tcp   open  imap3
443/tcp   open  https
445/tcp   open  microsoft-ds
465/tcp   open  smtps
993/tcp   open  imaps
995/tcp   open  pop3s
1023/tcp  open  netvenuechat
1025/tcp  open  NFS-or-IIS
2105/tcp  open  eklogin
3372/tcp  open  msdtc
5000/tcp  open  UPnP
```

```
10000/tcp open  snet-sensor-mgmt
17300/tcp open  kuang2

Nmap finished: 1 IP address (1 host up) scanned in 0.424 seconds
```

5.4 Implmentació i contrucció d'un Honeypot

La implementació del nostre propi honeypot, va sorgir de la curiositat de saber que succeeix a la DMZ. Així doncs, en primer lloc vam ficar una màquina nua per veure el que passava. No vam ser vulnerats, però es va registrar un intent de connexió al port 22 corresponent al sshd.

```
May 31 17:02:06 senbei sshd[9334]: Did not receive
identification string from 61.235.77.105
May 31 17:07:13 senbei sshd[9385]: Invalid user admin from
61.235.77.105
May 31 17:07:33 senbei sshd[9386]: input_userauth_request:
 invalid user admin
May 31 17:07:33 senbei sshd[9385]: pam_unix(sshd:auth): check
pass; user unknown
May 31 17:07:33 senbei sshd[9385]: pam_unix(sshd:auth):
authentication failure; logname= uid=0 euid=0 tty=ssh ruser=
rhost=61.235.77.105
May 31 17:07:33 senbei sshd[9385]: pam_succeed_if(sshd:auth):
error retrieving information about user admin
```

Això ens va cridar la atenció i vam decidir escoltar més ports utlitzant l'eina `netcat` per a la escolta de certs ports que ens van semblar interessants i per on podiem rebre proves. El codi de l'script es mostra en l'Annex I.

Els resultats van ser força curiosos, ja que vam trobar conexions al port 80 corresponent al httpd. Aquests resultats es mostren en l'Annex II.

Així que per últim vam decidir simular el funcionament d'un servidor Apache dins d'un entorn chroot al qual s'hi accedia per ssh. El login i password serien els provats en el log del sshd:

login: admin

password: admin

Per a la construcció de l'entorn de gàbia vàrem utilitzar l'eina *chrootssh*. Tot seguit descriurem pas a pas la construcció d'un entorn chroot. El propòsit de la construcció d'aquesta gàbia es què cada connexió ssh estigui controlada dins d'un directori que nosaltres decidirem. Per a simular un sistema el primer que farem serà copiar les llibreries necesàries:

```
/bin/sh
/bin/cp
/bin/false
/bin/ls
/bin/mv
/bin/pwd
/bin/rm
/bin/rmdir
/bin/sh
/bin/true
/etc/group
/etc/passwd
/lib/ld-linux.so.2
/lib/libacl.so.1
/lib/libattr.so.1
/lib/libc.so.6
/lib/libcom_err.so.2
/lib/libcrypt.so.1
```

```
/lib/libcrypto.so.4
/lib/libdl.so.2
/lib/libnsl.so.1
/lib/libpthread.so.0
/lib/libresolv.so.2
/lib/librt.so.1
/lib/libselinux.so.1
/lib/libtermcap.so.2
/lib/libutil.so.1
/usr/lib/libz.so.1
/usr/lib/libgssapi_krb5.so.2
/usr/lib/libk5crypto.so.3
/usr/lib/libkrb5.so.3
/usr/libexec/openssh/sftp-server
/sbin/nologin
```

En els fitxers `/etc/password` i `/etc/groups` hi haurà la següent informació dins la ruta relativa:

/etc/password: `admin:x:503:503::/home/admin:/bin/bash`

/etc/groups: `admin:x:503:`

Després l'usuari root haurà de modificar l'estructura de la gàbia i el seu contingut.

```
$chroot /home/chroot
```

Un cop construïda la gàbia instal·larem l'`httpd`.

Capítol 6

Anàlisi de les dades obtingudes

6.1 Intranet

Durant la captura de dades en aquesta intranet es van rebre aquest missatges del Nepenthes. Les següents línies només són una mostra de tot l'atac rebut, ja que en realitat es van rebre moltes més proves.

```
[16042007 18:33:16 dia] Stored Hexdump var/hexdumps/9e688c58a5487b8eaf69c9e
1005ad0bf.bin (0x08c83438 , 0x00000001).
[16042007 18:37:56 dia] Stored Hexdump var/hexdumps/9e688c58a5487b8eaf69c9e
1005ad0bf.bin (0x08c83438 , 0x00000001).
[16042007 18:56:51 dia] Stored Hexdump var/hexdumps/9e688c58a5487b8eaf69c9e
1005ad0bf.bin (0x08caed70 , 0x00000001).
[16042007 19:29:54 dia] Stored Hexdump var/hexdumps/9e688c58a5487b8eaf69c9e
1005ad0bf.bin (0x08c83438 , 0x00000001).
[16042007 20:01:26 dia] Stored Hexdump var/hexdumps/9e688c58a5487b8eaf69c9e
1005ad0bf.bin (0x08caed70 , 0x00000001).
[16042007 20:45:27 dia] Stored Hexdump var/hexdumps/9e688c58a5487b8eaf69c9e
1005ad0bf.bin (0x08c83438 , 0x00000001).
```

Nepenthes va ser incapaç de descobrir de quin virus es tractava, i tampoc reportant-ho a VirusTotal vam aconseguir saber-ho. Malgrat aquest problema, Nepenthes demostra la seva vàlua com element actiu de seguretat, ja que la infecció d'aquest virus va aconseguir col·lapsar alguns equips de xarxa degut al tràfic generat per a la seva dispersió.

L'administrador de xarxa no va assabentar-se de que alguna cosa succeïa fins que es va produïr un increment elevat del trànsit de la xarxa que va provocar l'alentiment en les comunicacions i que alguns equips de la xarxa tinguessin retards a la resposta a l'equip de supervisió.

Un cop detectat aquest problema es va haver d'actuar, localitzant i aïllant les màquines infectades en els diversos campus de la UdL.

Nepenthes va informar molt abans de l'actuació d'un virus, cosa que faria que l'atac no hagués comportat tants problemes al tràfic.

6.2 DMZ

El primer atac registrat a la DMZ va ser les proves al port 22. Són unes de les més típiques i intenten aprofitar-se de l'ús d'anti-claus, claus basades en diccionari i atacs per força bruta. L'objectiu és apoderar-se d'una *shell* remota per a l'execució de diferents programes.

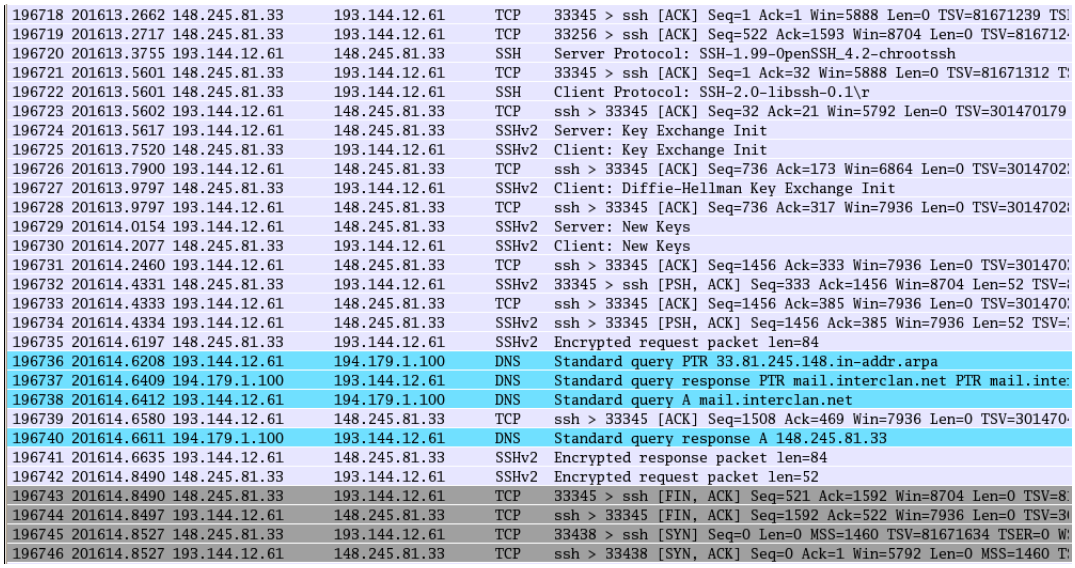
Uns altres típus d'atac molt típics són els rebuts al port 80, corresponent als servidors web. És aqui on la majoria de servidors allotjen les planes web, molt sovint utilitzant eines com: WordPress, Joomla, etc. Si aquestes eïnes no estan actualitzades poden ser object d'atac.

Es por veure en el següent log enregistrat en el port 80:

```
GET /w00tw00t.at.ISC.SANS.Pwn!t:) HTTP/1.1
```

Es tracta de la signatura d'un scanner de seguretat anomenat DFind. Com `netcat` no va retornar cap missatge l'atacant ho va deixar còrrer. Si haguessin trobat un servidor web real possiblement haguessin provat de fer alguna mena d'atac.

En l'intent de capturar informació utilitzant l'entorn chroot no es va poder enregistrar cap atac, en quant a malware o eines dels intrusos es refereix. Després d'una setmana d'observació el directori dedicat a la gàbia no es va modificar, tanmateix amb l'ajuda d'un sniffer es van detectectar nombroses connexions via SSH. Es va deduïr que hi havia moltes proves de connexió via



196718	201613.2662	148.245.81.33	193.144.12.61	TCP	33345 > ssh [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSV=81671239 TS
196719	201613.2717	148.245.81.33	193.144.12.61	TCP	33256 > ssh [ACK] Seq=522 Ack=1593 Win=8704 Len=0 TSV=816712
196720	201613.3755	193.144.12.61	148.245.81.33	SSH	Server Protocol: SSH-1.99-OpenSSH_4.2-chrootssh
196721	201613.5601	148.245.81.33	193.144.12.61	TCP	33345 > ssh [ACK] Seq=1 Ack=32 Win=5888 Len=0 TSV=81671312 T
196722	201613.5601	148.245.81.33	193.144.12.61	SSH	Client Protocol: SSH-2.0-libssh-0.1\r
196723	201613.5602	193.144.12.61	148.245.81.33	TCP	ssh > 33345 [ACK] Seq=32 Ack=21 Win=5792 Len=0 TSV=301470179
196724	201613.5617	193.144.12.61	148.245.81.33	SSHv2	Server: Key Exchange Init
196725	201613.7520	148.245.81.33	193.144.12.61	SSHv2	Client: Key Exchange Init
196726	201613.7900	193.144.12.61	148.245.81.33	TCP	ssh > 33345 [ACK] Seq=736 Ack=173 Win=6864 Len=0 TSV=3014702
196727	201613.9797	148.245.81.33	193.144.12.61	SSHv2	Client: Diffie-Hellman Key Exchange Init
196728	201613.9797	193.144.12.61	148.245.81.33	TCP	ssh > 33345 [ACK] Seq=736 Ack=317 Win=7936 Len=0 TSV=3014702
196729	201614.0154	193.144.12.61	148.245.81.33	SSHv2	Server: New Keys
196730	201614.2077	148.245.81.33	193.144.12.61	SSHv2	Client: New Keys
196731	201614.2460	193.144.12.61	148.245.81.33	TCP	ssh > 33345 [ACK] Seq=1456 Ack=333 Win=7936 Len=0 TSV=301470
196732	201614.4331	148.245.81.33	193.144.12.61	SSHv2	33345 > ssh [PSH, ACK] Seq=333 Ack=1456 Win=8704 Len=52 TSV=
196733	201614.4333	193.144.12.61	148.245.81.33	TCP	ssh > 33345 [ACK] Seq=1456 Ack=385 Win=7936 Len=0 TSV=301470
196734	201614.4334	193.144.12.61	148.245.81.33	SSHv2	ssh > 33345 [PSH, ACK] Seq=1456 Ack=385 Win=7936 Len=52 TSV=
196735	201614.6197	148.245.81.33	193.144.12.61	SSHv2	Encrypted request packet len=84
196736	201614.6208	193.144.12.61	194.179.1.100	DNS	Standard query PTR 33.81.245.148.in-addr.arpa
196737	201614.6409	194.179.1.100	193.144.12.61	DNS	Standard query response PTR mail.interclan.net PTR mail.inte
196738	201614.6412	193.144.12.61	194.179.1.100	DNS	Standard query A mail.interclan.net
196739	201614.6580	193.144.12.61	148.245.81.33	TCP	ssh > 33345 [ACK] Seq=1508 Ack=469 Win=7936 Len=0 TSV=301470
196740	201614.6611	194.179.1.100	193.144.12.61	DNS	Standard query response A 148.245.81.33
196741	201614.6635	193.144.12.61	148.245.81.33	SSHv2	Encrypted response packet len=84
196742	201614.8490	148.245.81.33	193.144.12.61	SSHv2	Encrypted request packet len=52
196743	201614.8490	148.245.81.33	193.144.12.61	TCP	33345 > ssh [FIN, ACK] Seq=521 Ack=1592 Win=8704 Len=0 TSV=8
196744	201614.8497	193.144.12.61	148.245.81.33	TCP	ssh > 33345 [FIN, ACK] Seq=1592 Ack=522 Win=7936 Len=0 TSV=3
196745	201614.8527	148.245.81.33	193.144.12.61	TCP	33438 > ssh [SYN] Seq=0 Len=0 MSS=1460 TSV=81671634 TSER=0 W
196746	201614.8527	193.144.12.61	148.245.81.33	TCP	ssh > 33438 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 T

Figura 6.1: Captura d'un dels molts intents de connexio SSH

ssh, però poques duraven lo suficient com per fer cap transferència de fitxers, això podria ser degut a que o bé no es va encertar la clau (cosa poc probable donada la simplicitat del login i password) o que l'intrús s'adonava de era un entorn de gàbia i seguia provant claus.

Al mateix temps que es rebien aquests atacs, Nepenthes va capturar dos intents d'atac als ports 22 i 80 on se simulava una línia de comandes remota DCOM i un servidor web IIS:

```
 Nepenthes Version 0.2.0
Compiled on Linux/x86 at Apr 12 2007 11:52:57 with g++ 4.1.1 20070105
(Red Hat 4.1.1-51)
Started on senbei running Linux/i686 release 2.6.20-1.2316.fc5
[ info mgr ] Loaded Nepenthes Configuration from
 "/opt/nepenthes/etc/nepenthes/nepenthes.conf".
[ info net mgr ] Using 193.144.12.62 as bind_address for all connections
[ info sc module ] Loading signatures from file
 var/cache/nepenthes/signatures/shellcode-signatures.sc
[ info mgr ] The process 18014 was given capabilities =
 cap_setgid,cap_setuid,cap_net_bind_service,cap_net_admin,cap_net_raw,
cap_sys_chroot+eip
```

```
[ warn dia ] Unknown IIS 11 bytes State 0
[ dia  ] Stored Hexdump var/hexdumps/cebfd9255328ac2fb852f1edb3eeb5ed.bin
 (0x0947c640 , 0x0000000b).
[ warn handler dia ] Unknown DCOM Shellcode (Buffer 0 bytes) (State 0)
[ handler dia ] Ignoring zero-length hexdump.
```

Ambdos avisos són de molt pocs bytes el que fa pensar que no vàren arribar a ser atacs, tot i així es poden catalogar com proves als ports o intents d'atac.

Capítol 7

Conclusions

Seguretat de les xarxes internes de la UdL

S'ha pogut comprovar durant la realització del treball, la dificultat de poder trobar forats de seguretat en les xarxes internes de la Universitat de Lleida: Intranet, Alumnes i Wifi. Malgrat que segurament l'observació de successos hauria d'haver estat una mica més perllongada. El fet que l'accés des de fora estigui salvaguardat per firewalls aporta un plus de seguretat. A més, les restriccions fetes als usuaris en l'autenticació com en l'ús del programari ajuda a què el tràfic no sigui tant descontrolat. És per això, que en l'observació feta en aquestes VLAN només s'han trobat atacs corresponents a la propagació de virus. Aquest tipus d'atacs són pràcticament incontrolables ja que provenen de fonts alienes a l'administració de la xarxa donat que molts equips d'usuaris, com poden ser els portatils, és connecten en diverses xarxes diferents a la de la UdL.

Nepenthes ha demostrat la seva vàlua com a element de seguretat informant abans que ningú de l'activitat maliciosa en la xarxa.

Inseguretat i proves constants en les xarxes accessibles des d'Internet

L'observació continuada a la DMZ, ha estat la més enriquidora, ja que la captura de diferents atacs provinents d'arreu han constatat lo exposats al

perill que estan els servidors en aquesta VLAN. La vàlua dels honeypots en aquesta zona no recau tant en l'alerta, sinò en l'aprenetatge de tècniques i procediments que els intrusos utilitzen en els atacs. Així doncs hem pogut veure com el port 80 ha estat el punt de mira de molts atacs, el que ens ha demostrat que un dels talons d'Aquiles dels nostres servidors externs són les aplicacions web. El fet de mantenir actualitzada la nostra aplicació web esdevé un punt clau en la defensa contra intrusions al sistema, ja que constantment serem provats en la recerca de fallides de codi o *backdoors*.

També hem estat objecte d'atacs al port 22 corresponent al dimoni *sshd* Els logs ens han demostrat la importància d'una bona clau per a l'entrada al nostre sistema. Les claus basades en diccionari o les més típiques conformen el diccionari dels intrusos. Aquest diccionari serà utilitzat per intentar entrar en algun sistema d'una manera ràpida i senzilla, però que pot comportar la pèrdua del nostre servidor envers als atacants.

Versatilitat dels Honeypot com a eina d'anàlisi

L'incís en el funcionament dels honeypot ha estat essencial per a la comprovació de la seva versatilitat. D'aquesta manera trobem una sol·lució per a cada problema. L'únic que definex la tecnologia honeypot és la filosofia de l'engany per a passar d'un estat d'espera als atacs de fora, a una postura activa que comporta la lluita contra els atacants dins dels honeypots. La construcció de cada honeypot dependrà dels objectius que es vulguin assolir, variant la interactivitat i el rol. Durant la construcció, tot s'hi val, l'ingeni predomina i el domini de les eines necessàries esdevé la columna vertebral del honeypot.

Éxit del Software i Coneixement Lliure com a aportació a comunitat informàtica contra els *black hats*

Sense una filosofía oberta al coneixement, la guerra contra els atacants està perduda. Què hagués passat si el primer honeypot hagués estat dissenyat sota el màxim secret? Què passaria si els descobriments de nous atacs i

programari maliciós fossin emmagatzemats per a l'ús d'una sola companyia? La resposta és senzilla: tots seriem vulnerats permanentment.

El coneixement de nous atacs i noves tècniques ens fa forts contra tot allò que prové d'Internet. Gràcies a això podem prendre la iniciativa contra els intrusos i tractar-los de tu a tu. La divulgació del software lliure, ens permet la millora dels nostres sistemes i de la seguretat de les nostres xarxes, que conformen la gran xarxa de xarxes. Per tant, inherentment contribuïm a la seguretat d'Internet.

Capítol 8

Treball futur

Aquest treball només ha estat una anàlisi de la seguretat en la xarxa de la UdL. S'ha demostrat que mitjançant l'ús d'aquestes eines es pot contribuir a la millora de la seguretat i per tant al rendiment de la xarxa. Una continuació d'aquest treball podria ser el desenvolupament d'un honeypot específic per a la xarxa de la UdL, que es mantingués permanement des de l'administració de la xarxa.

Com s'ha pogut veure també, la DMZ de la UdL és un escenari ideal per a la captura de nous atacs i altres intrusions. La implementació de honeypots de recerca o honeynets en aquesta VLAN podria proporcionar molta informació, no només per ser aprofitat dins de la UdL, sinò com a aportació a la comunitat.

És evident que projectes com Honeyfarms o Honeynets distribudes, esdevindran en un futur pròxim la base del coneixement de la seguretat a Internet. Així doncs la col·laboració en aquesta mena de projectes poden ser molt enriquidores per a les institucions que basen el seu funcionament en les xarxes de comunicacions.

Apèndix A

port-listener.sh

```
#!/bin/bash

/etc/init.d/sshd stop
/etc/init.d/sendmail stop

nc -lk 80 > port80.log
nc -lk 21 > port21.log
nc -lk 22 > port22.log
nc -lk 23 > port23.log
nc -lk 25 > port25.log
nc -lk 42 > port42.log
nc -lk 109 > port109.log
nc -lk 110 > port110.log
nc -lk 137 > port137.log
nc -lk 138 > port138.log
nc -lk 139 > port139.log
nc -lk 143 > port143.log
nc -lk 161 > port161.log
nc -lk 194 > port194.log
```

Apèndix B

Atac al port 80

```
GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows 98)
Host: 193.144.12.61
Connection: Keep-Alive

GET /w00tw00t.at.ISC.SANS.Pwn!t:) HTTP/1.1

GET / HTTP/1.1
Host: 193.144.12.61
Connection: keep-alive
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows 98)
X-Forwarded-For: 10.33.8.88
Via: 1.1 nc-corpse (NetCache NetApp/6.0.3)

GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows 98)
Host: 193.144.12.61
```

```
Connection: Keep-Alive

GET /w00tw00t.at.ISC.SANS.DFind:) HTTP/1.1

GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows 98)
Host: 193.144.12.61
Connection: Keep-Alive

GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows 98)
Host: 193.144.12.61
Connection: Keep-Alive

GET / HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 Fedora/1.5.0.10-1.fc5 Firefox/1.5.0.10
Accept: text/xml,application/xml,application/xhtml+xml,text/html;
q= 0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
```

Bibliografia

[1] Baecher, P. , Koetter, M. , Holz, T. , Dornsief, M. , Freiling, F. "The Nepenthes Platform: An Efficient Approach to Colect Malware". 2006.

[2] Chandran, R. , Pakala, S. "Simulating Networks with Honeyd". 2003.

[3] Cohen, F. "The Decption Toolkit". 1998.

[4] Comer, D. E. "Redes globales de información con Internet y TCP/IP". Pentice Hall, 1996.

[5] Corletti, A. "Nivel de Inmadurez de los Sistemas de Detección de Intrusiones de Red (NIDS)". 2001.

[6] Garfinkel, S. , Spafford, G. , "Practical UNIX & Internet security". O'Reilly, 1991.

[7] Gubbels, K. "Hands in the Honeypot". GIAC Security Essentials Certification, 2002.

[8] Halsall, F. "Comunicación de datos, redes de computadores y sistemas abiertos". Addison-Wesley, cuarta edición, 1996.

[9] Hernández, M. J. , Lerna, C. F. "Aplicaciones Prácticas de los Honeypots en la Protección y Monitoreo de Redes de Información".

[10] Provos, N. "A Virtual Honeypot Framework". In *Proceedings of 13th USENIX Security Symposium*, pages 1-14, 2004.

[11] The Honeynet Project. "Know Your Enemy". 2000.

[12] The Honeynet Project. “Sebek: A Kernel based data capture tool”. 2003.

[13] Kalle Dalheimer, M. , Kaufman , L. Welsh, M. “Running Linux”. O'-Reilly, 3a ed, 1999.

[14] León-García, A., Widjaja, I. “Redes de comunicación”. McGraw Hill, 2002.

[15] Márquez Gárcia, Fco. Manuel. “UNIX. Programación avanzada”. Rama, 1993.

[16] McMullen, J.F. “Enahance Intrusion Detection with a Honeypot”. TechRepublic, 2001.

[17] Spitzner, L.“Honeypots: Tracking Hackers”. Addison Wesley, 2002.

[18] Stallings, W. “Comunicaciones y redes de computadores”. Pentice Hall, 7ª ed., 2004.

[19] Stevens, W. R. “TCP/IP Illustrated volume 1”. Addison Wesley, 1994.

[20] Tackett, J. , Gunter, D. “Utilizando Linux”. Prentice Hall, 2a ed., 1996.

[21] Tanembaum, A.S. , “Redes de computadores”. Prentice Hall, 4ª ed., 2003.